

# SBC-GPC v2.00 PROGRAMMERS GUIDE



SBC-GPC v2.00 Programmers Guide

Part Number 1E-04-00-0146

© 2012 American Auto-Matrix™

This document is protected by copyright and is the property of American Auto-Matrix. It may not be used or copied in whole or in part for any purpose other than that for which it is supplied without authorization. This document does not constitute any warranty, expressed or implied.

Every effort has been made to ensure that all information was correct at the time of publication. American Auto-Matrix reserves the right to alter specifications, performance, capabilities and presentation of this product at any time.

American Auto-Matrix and Auto-Matrix are trademarks of American Auto-Matrix and are not to be used for publication without the written consent of American Auto-Matrix.

All other brand names or product names are trademarks or registered trademarks of their respective companies or organizations.

## WORLD HEADQUARTERS

American Auto-Matrix  
One Technology Lane  
Export, Pennsylvania 15632-8903 USA  
Tel (1) 724-733-2000  
Fax (1) 724-327-6124  
Email [aam@aamatrix.com](mailto:aam@aamatrix.com)  
[www.aamatrix.com](http://www.aamatrix.com)

**Updated 10/8/2012**

- . Section 7 Added attribute ;P0 for control loops
- . Section 9.9 Timers Channel added
- . Added Section 6.19.2 - Writing to Text-Based Attribute Values

**Updated 1/04/2011**

- . Initial Release - Corresponds to v2.00 Firmware Release

**Updated 3/02/2011**

- . Corresponds with v2.01.05 Firmware Release
- . Section 9.2 Added attributes F31x;GT, GE, LT, LE, ET Boolean Variables for Math Channel
- .

**Updated 4/28/2011**

- . Section 9.8.2.1 Removed attributes F40x;SU,SL for Delay On/Off modes, clarified P1 and P2 as unload/load setpoints
- . Section 8.5 Occupancy Extension attribute descriptions updated
- .

**Updated 6/27/2011**

- . Added Appendices A, B and C
- . Miscellaneous grammar corrections
- .

**Updated 12/27/2011**

- . Section 8 - Added attributes F90x; SD,FB,CU,CW,CO,CS,SO and SF
- . Updated Appendix A to include added F90x attributes



This manual describes the operation and configuration of SBC-GPC product family firmware v2.00 and later.

This document is divided into the following sections:

- . One: *Overview*, describing the product platform.
- . Two: *Device Setup*, describing the setup and configuration of device and communications.
- . Three: *Inputs Setup*, describing the setup and configuration of inputs.
- . Four: *Outputs Setup*, describing the setup and configuration of outputs.
- . Five: *Expansion I/O*, describing the setup and configuration of expandable STATbus I/O modules.
- . Six: *Programs and Files*, describing SPL programming.
- . Seven: *Control Loop*, describing how to use and implement control loops.
- . Eight: *Scheduling*, describing how to setup and configure schedules and holidays.
- . Nine: *Data Manipulation*, describes data manipulation objects.
- . Ten: *Data Movement and Storage*, describes data movement objects.
- . Eleven: *Special Modes*, describes the use of special mode objects.

This document contains certain style and formatting conventions for conveying information in a clear and concise manner:

- . Menu commands appear with a ">" symbol between levels. For example: File>Open.
- . *Italics* indicate options within software.



---

1.1 What is SBC-GPC? .....	1-3
1.2 Fundamental Concepts.....	1-4
1.2.1 American Auto-Matrix PUP Protocol.....	1-4
1.2.2 Token Passing .....	1-4
1.2.3 Token Error Recovery.....	1-5
1.3 PUP Data Structure .....	1-7
1.3.1 Channels.....	1-7
1.3.2 Attributes.....	1-7
2.1 System Overview.....	2-3
2.2 Device Address Configuration .....	2-4
2.3 Token Passing .....	2-5
2.4 Time and Date Configuration.....	2-6
2.4.1 Time Synchronization .....	2-6
2.4.2 Daylight Saving .....	2-7
3.1 Inputs Overview .....	3-3
3.1.1 Universal Input Overview .....	3-3
3.1.2 Digital Inputs Overview .....	3-3
3.1.3 Programming Concepts and Techniques.....	3-3
3.2 Universal Inputs.....	3-5
3.2.1 IVR Jumpers for Universal Inputs .....	3-5
3.2.2 Connecting Universal Inputs .....	3-6
3.2.3 Universal Input Configuration.....	3-7
3.2.4 Configuring Analog Input Alarm/Event Notifications .....	3-11
3.2.5 Digital Input Configuration.....	3-11
3.2.6 Using Universal Inputs for Pulse Counting Applications .....	3-12
3.2.7 Configuring Digital Input Alarm/Event Notifications .....	3-13
3.3 Digital Inputs .....	3-14
3.3.1 Connecting Digital Inputs .....	3-14
3.3.2 Configuring the Digital Inputs.....	3-15
3.4 Piecewise Curves .....	3-16
3.4.1 Overview of X and Y .....	3-16
3.4.2 Piecewise Curves for Voltage Inputs .....	3-16
3.4.3 Piecewise Curves for Current Inputs .....	3-18
3.4.4 Piecewise Curves for Resistance Inputs.....	3-18
4.1 Outputs Overview .....	4-3
4.1.1 Analog Output Overview .....	4-3
4.1.2 Digital Output Overview .....	4-3
4.1.3 Programming Concepts and Techniques.....	4-3
4.2 Analog Output Summary .....	4-4
4.3 Analog Outputs .....	4-5
4.3.1 Configuring Minimum and Maximum Thresholds.....	4-5
4.3.2 Control Mode .....	4-5
4.3.3 Input Interlocking.....	4-5
4.3.4 Additional Interlock.....	4-5
4.3.5 Actual Output Voltage and Actual Output Current .....	4-5

4.3.6 Communication Failure Enable Positioning .....	4-6
4.3.7 Fire Override .....	4-6
4.3.8 Update Threshold.....	4-6
4.3.9 Run Hours .....	4-6
4.4 Digital Output Summary.....	4-7
4.5 Digital Outputs .....	4-9
4.5.1 Configuring Minimum Off/On Times.....	4-9
4.5.2 Configuring Polarity.....	4-9
4.5.3 Actual Output State .....	4-9
4.5.4 Control Mode.....	4-9
4.5.5 Input Interlocking.....	4-9
4.5.6 Additional Interlock.....	4-9
4.5.7 Schedules to Follow .....	4-10
4.5.8 Inrush Current Lockout.....	4-10
4.5.9 Pulse Width .....	4-10
4.5.10 Run Hours .....	4-10
5.1 What are IOX Modules?.....	5-3
5.1.1 Features of IOX Modules .....	5-3
5.1.2 Remote I/O and Mapping Points .....	5-3
5.2 IOX Module Specifications.....	5-4
5.2.1 General .....	5-4
5.2.2 SSB-FI1.....	5-4
5.2.3 SSB-UI1 .....	5-4
5.2.4 SSB-AO1.....	5-4
5.2.5 SSB-DI1 .....	5-5
5.2.6 SSB-DO1 .....	5-5
5.2.7 SSB-DO1-I .....	5-5
5.2.8 SSB-DO2 .....	5-5
5.2.9 SSB-DO2-I .....	5-5
5.2.10 SSB-IOX1-1 .....	5-6
5.2.11 SSB-IOX1-2 .....	5-6
5.2.12 SSB-IOX2-1 .....	5-6
5.2.13 SSB-IOX2-2 .....	5-6
5.2.14 SSB-STAT1D, SBC-STAT2D, SBC-STAT3.....	5-6
5.2.15 SBC-RH1, SBC-RH3.....	5-7
5.2.16 SBC-RHT .....	5-7
5.3 Length of the Network.....	5-8
5.4 Number of Devices .....	5-9
5.4.1 Communications Limits .....	5-9
5.5 GID Numbers and Mapping IOX Modules .....	5-11
5.5.1 Writing GIDs to Devices .....	5-11
5.5.2 Removing GID assignments .....	5-11
5.6 SSB-FI1 .....	5-13
5.6.1 Features.....	5-13
5.6.2 Wiring/Configuration.....	5-13

---

5.6.3 Mounting the SSB-FI1 .....	5-16
5.6.4 Status Indicator LED .....	5-16
5.6.5 SSB-FI Configuration Table .....	5-17
5.7 SSB-UI1 .....	5-18
5.7.1 Features .....	5-18
5.7.2 Wiring/Configuration .....	5-18
5.7.3 Mounting the SSB-UI1 .....	5-22
5.7.4 Status Indicator LED .....	5-23
5.7.5 SSB-UI Configuration Table .....	5-24
5.8 SSB-AO1 .....	5-25
5.8.1 Features .....	5-25
5.8.2 Wiring/Configuration .....	5-25
5.8.3 Mounting the SSB-AO1 .....	5-28
5.8.4 Status Indicator LED .....	5-29
5.8.5 SSB-AO Configuration Table .....	5-30
5.9 SSB-DI1 .....	5-31
5.9.1 Features .....	5-31
5.9.2 Wiring/Configuration .....	5-31
5.9.3 Mounting the SSB-DI1 .....	5-33
5.9.4 Status Indicator LED .....	5-34
5.9.5 SSB-DI1 Configuration Table .....	5-34
5.10 SSB-DO1 .....	5-36
5.10.1 Features .....	5-36
5.10.2 Mounting the SSB-DO1 .....	5-36
5.10.3 Wiring/Configuration .....	5-37
5.10.4 SSB-DO1 Configuration Table .....	5-39
5.11 SSB-DO1-I .....	5-40
5.11.1 Features .....	5-40
5.11.2 Mounting the SSB-DO1-I .....	5-40
5.11.3 Wiring/Configuration .....	5-41
5.11.4 SSB-DO1-I Configuration Table .....	5-44
5.12 SSB-DO2 .....	5-45
5.12.1 Features .....	5-45
5.12.2 Mounting the SSB-DO2 .....	5-45
5.12.3 Wiring/Configuration .....	5-46
5.12.4 SSB-DO2 Configuration Table .....	5-48
5.13 SSB-DO2-I .....	5-49
5.13.1 Features .....	5-49
5.13.2 Mounting the SSB-DO2-I .....	5-49
5.13.3 Wiring/Configuration .....	5-50
5.13.4 SSB-DO1-I Configuration Table .....	5-53
5.14 SSB-IOX Family .....	5-54
5.15 SSB-IOX1-x .....	5-55
5.15.1 SSB-IOX1-1 Features .....	5-55
5.15.2 SSB-IOX1-2 Features .....	5-55

5.15.3 Wiring/Configuration.....	5-56
5.15.4 Network & Power .....	5-56
5.15.5 Universal Inputs .....	5-56
5.15.6 Digital Inputs .....	5-58
5.15.7 Analog Outputs .....	5-59
5.15.8 Digital Outputs.....	5-60
5.15.9 Mounting the SSB-IOX1-X .....	5-61
5.15.10 Status Indicator LED .....	5-62
5.15.11 SSB-IOX1-1 Configuration Table .....	5-63
5.16 SSB-IOX2-x .....	5-64
5.16.1 SSB-IOX2-1 Features .....	5-64
5.16.2 SSB-IOX2-2 Module.....	5-64
5.16.3 Wiring/Configuration.....	5-65
5.16.4 Network & Power .....	5-65
5.16.5 Universal Inputs .....	5-66
5.16.6 Analog Outputs .....	5-69
5.16.7 Digital Outputs.....	5-70
5.16.8 Mounting the SSB-IOX2-X .....	5-71
5.16.9 Status Indicator LED .....	5-71
5.16.10 SSB-IOX2-1 Configuration Table .....	5-72
6.1 Overview .....	6-3
6.2 SPL Programming.....	6-4
6.2.1 Loading Programs into GPC .....	6-4
6.3 Introduction to SPL .....	6-5
6.4 The Parts of SPL Programs .....	6-6
6.5 Program Names.....	6-7
6.6 The .SPL, .PLB and .LST Files .....	6-8
6.7 Attributes and Registers.....	6-9
6.8 Compiler Control Statements.....	6-10
6.9 Comments.....	6-12
6.10 Labels .....	6-13
6.11 Expressions .....	6-14
6.12 Program Statements Overview .....	6-16
6.13 Assignment Statements and Equates.....	6-18
6.13.1 Standard Value assignment .....	6-18
6.13.2 EQU .....	6-19
6.14 Iteration, Branching and Subroutines.....	6-20
6.14.1 GOTO statement.....	6-20
6.14.2 IF... THEN... {ELSE...} Statement .....	6-20
6.14.3 ON... GOTO... statement .....	6-21
6.14.4 LOOP Statement.....	6-21
6.14.5 GOSUB Statement.....	6-22
6.14.6 RETURN Statement.....	6-22
6.15 Program Delays .....	6-23
6.15.1 SWAIT and MWAIT Statements.....	6-23

---

6.15.2 WAIT Statement.....	6-23
6.16 Execution Error Control .....	6-24
6.16.1 ERRORABORT Statement .....	6-24
6.16.2 ERRORWAIT Statement.....	6-24
6.16.3 ONERROR Statement .....	6-24
6.17 Debugging Statements .....	6-26
6.17.1 SECTION Statement.....	6-26
6.18 Program Control Attributes .....	6-27
6.19 Working with Channel Attributes.....	6-31
6.19.1 Addressing Channel Attributes .....	6-31
6.19.2 Addressing User-Defined Attributes.....	6-32
6.19.3 Peer-To-Peer Addressing .....	6-32
7.1 Control Loops Overview .....	7-3
7.1.1 Programming Concepts and Techniques.....	7-3
7.2 Analog Output Control Loops .....	7-4
7.2.1 Basic Setup.....	7-4
7.2.2 Proportional Control Setup.....	7-5
7.2.3 Deadband Configuration .....	7-6
7.2.4 Reset Control Setup.....	7-8
7.2.5 Soft Start Setup.....	7-11
7.2.6 Enabling the Control Loop .....	7-11
7.3 Floating Point Control .....	7-12
7.3.1 Basic Setup.....	7-12
7.3.2 Proportional Control Setup.....	7-13
7.3.3 Deadband Configuration .....	7-14
7.3.4 Reset Control Setup.....	7-16
7.3.5 Calibration.....	7-19
7.3.6 Interlocking and Fire Positioning.....	7-21
7.3.7 Enabling the Control Loop .....	7-22
7.4 Thermostatic Control .....	7-23
7.4.1 Basic Setup.....	7-23
7.4.2 Enabling the Control Loop .....	7-24
8.1 Scheduling Overview .....	8-3
8.1.1 Schedule Summary.....	8-3
8.2 Main Schedule .....	8-4
8.3 Host Schedule .....	8-5
8.4 Broadcast Schedule.....	8-6
8.5 Occupancy Extension .....	8-7
8.6 Local Schedules .....	8-8
8.6.1 Operation .....	8-8
8.6.2 Schedule Example .....	8-10
8.6.3 Feedback Items .....	8-11
8.7 Holidays .....	8-12
8.7.1 Holiday Feedback .....	8-12
9.1 Data Manipulation Overview.....	9-3

9.1.1 Programming Concepts and Techniques.....	9-3
9.2 Math.....	9-4
9.3 Logic.....	9-5
9.4 Min/Max/Avg.....	9-6
9.5 Enthalpy.....	9-7
9.6 Scale.....	9-8
9.7 Input Select.....	9-9
9.8 Staging.....	9-10
9.8.1 Basic Configuration.....	9-10
9.8.2 Staging Modes.....	9-11
9.8.3 Stage Interlocking.....	9-11
10.1 Data Movement Overview.....	10-3
10.1.1 Programming Concepts and Techniques.....	10-3
10.2 Broadcasts.....	10-4
10.2.1 Broadcasting Concepts.....	10-4
10.2.2 Sending a Broadcast.....	10-4
10.2.3 Receiving a Broadcast.....	10-4
10.2.4 Feedback and Status Information.....	10-4
10.3 Persisted/Remap Values.....	10-5
10.3.1 Configure the Operating Mode.....	10-5
10.3.2 Defining Inputs and Outputs.....	10-5
10.3.3 Configuring the Triggering Mode and Input.....	10-5
10.3.4 Data Coercion Mode.....	10-5
10.3.5 Feedback Status.....	10-5
10.4 Alarm Setup.....	10-6
11.1 Summary of special modes.....	11-3
11.2 Fire Mode.....	11-4
11.3 Comm Failure.....	11-5

---

# SECTION 1: OVERVIEW

*This section provides general overview information regarding the SBC-GPC, as well as reviews fundamental concepts relative to the American Auto-Matrix PUP Protocol, and PUP Data Structure.*

## IN THIS SECTION

What is SBC-GPC?.....	1-3
Fundamental Concepts .....	1-4
American Auto-Matrix PUP Protocol .....	1-4
Token Passing.....	1-4
Token Error Recovery .....	1-5
PUP Data Structure.....	1-7
Channels.....	1-7
Attributes .....	1-7



## 1.1 WHAT IS SBC-GPC?

The SBC-GPC Product Family is a group of programmable building automation controllers, designed to provide a completely programmable solution for any control sequence encountered. GPC controllers are designed to provide hardware and software flexibility using on-board I/O and STATbus - AAM's innovative sensor networking technology.

SBC-GPC controllers can be used in a wide variety of applications that require stand-alone control, as well as full peer-to-peer network capabilities with other PUP devices. SBC-GPC product models are available in the following models:

- . SBC-GPC1
- . SBC-GPC2
- . SBC-GPC3

Because the GPC is a common platform, each product model is programmed exactly the same. There are obvious differences with each GPC controller - such as I/O count, object count, and I/O expansion capabilities. For more details on each GPC controller model, please reference the marketing data sheet for the model you are working with.

## 1.2 FUNDAMENTAL CONCEPTS

This section of the user manual reviews standard fundamental concepts and provides an explanation of the prerequisite information necessary to know prior to installing American Auto-Matrix SBC-Series products.

### 1.2.1 AMERICAN AUTO-MATRIX PUP PROTOCOL

American Auto-Matrix PUP is a half-duplex EIA-485 network, designed to connect multiple SBC-series controllers together in a multi-drop (daisy chained) topology. All network communications occurs at 9.6kbps, unless modified by user intervention to a higher speed. A PUP network can contain up to a maximum of 64 devices connected at a given time.

In many cases, an installed network controller or area controller solution is a full administrator with configurable token passing down to devices. In this situation, all controllers connected to it are configured as irresponsible peers.

In network topologies that do not require a network controller or area controller solution, you may elect to configure a unitary controller as a full administrator. In this type of topology, a unitary controller can pass the token to other devices that may need to access the network for application purposes (peer-to-peer SPL applications, temperature broadcasts, etc.)

### 1.2.2 TOKEN PASSING

SBC-GPC products are designed to maintain a master/slave relationship with other controllers on the same PUP network. The SBC-GPC may also be configured to operate as a peer with other PUP devices such as SBC-SD, SBC-ASC, AspectFT-Matrix Area Controller, and other such devices.

Peer-to-peer communication operates via token passing. Token passing is a communications scheme that allows PUP devices connected in a network to communicate with each other and share data among themselves. A "token" is passed from unit to unit on the network in a round-robin fashion. When any unit possesses the token, it performs any network activities for which it is responsible, and then passes the token to another unit. At any time, the unit that possesses the token is the only device permitted to initiate communications with another device on the network or to request information from it.

A device that receives the token may or may not need to perform network functions (e.g. read values from a remote device, broadcast information, etc.). If not, it will simply pass the token along the network. Determining how the token is passed along the network depends on which units may possess the token and what type of peer they are.

There are two types of devices that can potentially be connected on a PUP network:

- . Full Administrator
- . Irresponsible Peer

#### 1.2.2.1 FULL ADMINISTRATOR

A full administrator is a device that passes network tokens to extend PUP communications. A full administrator maintains a list of peers (irresponsible peers or other full administrators) that each get the token passed to them in a round-robin fashion. Peers are addressed by referencing the Unit ID of the device that will receive the network token. The SBC-GPC is capable of passing a network token to 12 peers.

A token is generally passed to a peer device with the understanding that the device that will receive the token will perform network transactions (read/write, broadcast, etc.). Once the peer device has finished

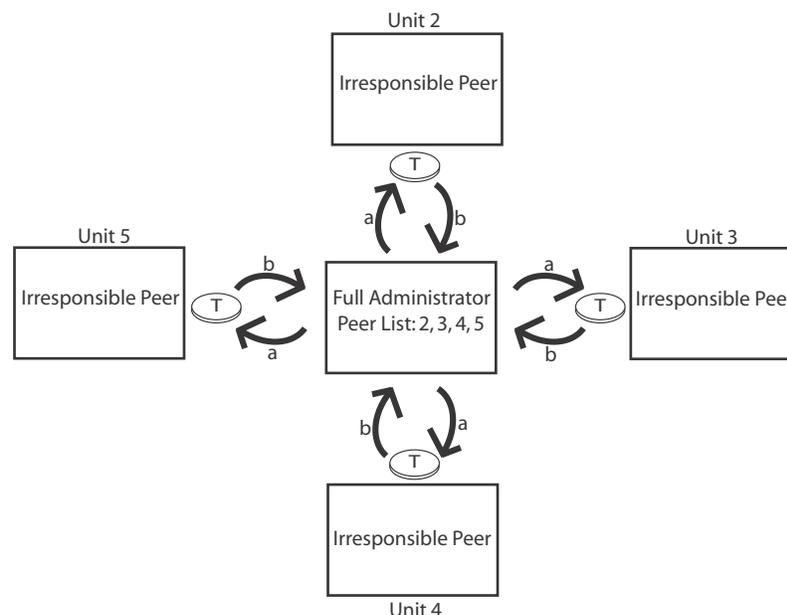
using the token, it will be returned back to the source full administrator. A unit is configured as a full administrator by setting FF00; (TP) **Token Passing Type** to a value of 1.

### 1.2.2.2 IRRESPONSIBLE PEER

An irresponsible peer is a device that can only return a network token to the full administrator that initially passed it the token. If an irresponsible peer is configured for an application that requires the network token, a full-administrator must pass the token to the irresponsible peer, through the use of its peer list.

### 1.2.2.3 NON-PEER DEVICES

While the SBC-GPC cannot be configured as a “non-peer”, it is necessary to understand the existence of non-peer devices and how they react in a PUP environment. Some legacy SOLO devices (SOLO/TX version 1.xx, SOLO/HX v1.xx, SOLO/MX v1.xx, etc.) are non-peers. Non-peers are devices that were not intended to use a network token. Unlike a full administrator (which passes the token to other units that are specified in its peer list) or an irresponsible peer (which passes the token back to its sender), a non-peer has no knowledge of token passing and will drop the token if it receives one. Non-peers may exist on token passing networks - but should not be passed a network token. A configuration in such a manner will result in repeated token drops, thereby reducing the performance of your network communications and other control factors.



a = send token to peer  
b = receive token back from peer

NOTE: The above illustration is a logical example, and not a physical network wiring example

Figure 1-1 Logic Example of PUP Network Token Passing

### 1.2.3 TOKEN ERROR RECOVERY

Because units can only perform network activities when they possess the token, it is important for the system to be able to recover the token if it is ever dropped or not passed. Full administrators are equipped with error recovery routines that allow them to “pick up” the token whenever it is dropped. Most devices capable of being configured as a full administrator have an FF00;ER parameter to enable token error recovery.

By default, the SBC-GPC will always perform error recovery when configured as a full administrator. Therefore, setting such a parameter is not necessary.

## 1.3 PUP DATA STRUCTURE

The SBC-GPC performs control and is programmed to do so through parameters located in memory. These parameters can be accessed and changed using SoloPro.

### 1.3.1 CHANNELS

All PUP-based devices have memory locations that specify details of how a particular feature of a given system operates. These memory locations, known as Channels, can be accessed through engineering software such as SoloPro.

Within a PUP device, there are two types of channels - Root Channels and Sub-Channels.

- Root channels are displayed with their full hexadecimal channel number (e.g. FE00), typically ending 00 as part of the hexadecimal location.
- Subchannels are indicated with the last two characters of the channel number. For example, subchannel 1 would be addressed as FE01; subchannel 2 as FE02, etc. There can be up to 256 subchannels (00-FFh) for every PUP channel.

Root Channel FE00 - Universal Inputs
FE01 - Universal Input 1
FE02 - Universal Input 2
FE03 - Universal Input 3
FE04 - Universal Input 4
FE05 - Universal Input 5

*Figure 1-2 Channel Example*

### 1.3.2 ATTRIBUTES

Within each channel or subchannel are sets of parameters, dictating how features associated to a portion of the device may react. These parameters, known as Attributes, provide representational data to a user regarding how features are handled. Attributes are identified by a case sensitive two-character name; generally a mnemonic for the functionality that it represents. A good example of an attribute is a current value interpreted by a Universal Input. An attribute known as “**CV**” would represent the current value for the Universal Input channel.

Channel FE01
;ON - Object Name = OAT
;CV - Current Value = 72.0
;RE - Reliability = 0 (Reliable)
;ST - Sensor Type = 7 (Precon Type 3)

*Figure 1-3 Channel and Attribute Example*

- When discussing attribute functions and their operational relationships, attributes are displayed following the channel to which they belong. For example, the channel FF00 controller manufacturer attribute (**CM**) is shown as FF00;**CM**. All attributes appear in boldface to aid in locating descriptions of a particular attribute in the text of this document. Descriptions of the SBC-SD channels and their attributes are contained in the following sections of this manual.

---

## SECTION 2: SYSTEM SETUP

*This section describes the process of system setup of SBC-GPC Products including device address, network communications, and time master options.*

### IN THIS SECTION

System Overview .....	2-3
Device Address Configuration.....	2-4
Token Passing.....	2-5
Time and Date Configuration .....	2-6
Time Synchronization.....	2-6
Daylight Saving .....	2-7



## 2.1 SYSTEM OVERVIEW

The general setup and configuration of a device is commonly achieved through accessing the **FF00** channel, which commonly contains system parameters. Within the SBC-GPC platform, each device supports a single FF00 channel which contains information regarding software/firmware loaded onto the device, as well as addressing and baud rate settings to allow the device to communicate on the network.

By default, all SBC-GPC products are shipped with a default configured network baud rate and device addressing. The default information is:

- . Baud Rate: pre-configured for 9.6kbps speed
- . Unit ID: factory configured to the last four (4) digits of the device serial number.

## 2.2 DEVICE ADDRESS CONFIGURATION

To configure the device address of an SBC-GPC, perform the following:

1. Discover the device and open it for monitoring using SoloPro Engineering Software
2. Select the System tab, containing **FF00** information
3. Configure the properties specifies below.

*Table 2-1: Device Address Configuration Properties*

Attribute	Valid Range	Notes
<b>(ID) Unit ID</b>	0 - 32768	<p>is the Unit ID Address; assigned to a PUP device to allow membership into the network. This number must be unique and cannot conflict with any other device connected on the local EIA-485 network.</p> <p>Changes made to this attribute are performed on-the-fly, therefore, you will be required to re-discover the device within SoloPro immediately afterwards to regain network communications.</p>
<b>(CP) Communication Speed</b>	0 = 9.6 kbps 6 = 38.4 kbps 7 = 19.2 kbps 8 = 115.2 kbps 9 = 57.6 kbps	<p>is the serial baud rate speed setting for the GPC. You must configure all devices on the same local EIA-485 network for the same baud. Mis-matched baud rates may result in communication failures.</p> <p>Changes made to this attribute are not established until the power has been reset via power-cycle or by writing to the <b>(RS) Reset Device</b> attribute.</p>

**CAUTION**



Traditionally, the prevailing baud rate of some PUP networks is 9.6kbps, as older PUP devices could not support PC baud rates any faster. If you are applying this product to an existing network with legacy products, carefully ensure that your existing devices will support faster speeds in the event you intend for your network to communicate faster.

## 2.3 TOKEN PASSING

SBC-GPC products are capable of acting as a Full Administrator, or Irresponsible Peer. As a full administrator, the SBC-GPC is capable of creating network tokens and passing them onto other devices on the local EIA-485 network. As an irresponsible peer, it is capable of receiving a network token from another device (configured as a Full Administrator) and uses the token to solicit network commands (such as peer-to-peer writes via SPL, or specialized broadcast using Broadcast objects).

In many situations where an area controller or small display operator interface is installed, the area controller or small display is commonly a Full Administrator and can be configured to pass network tokens to up to twelve (12) other PUP devices. Therefore, SBC-GPC products default as an irresponsible peer, dictated through the **(TP) Token Passing Type** attribute.

If you elect to configure the device as a full administrator:

1. Discover the device and open it for monitoring using SoloPro Engineering Software.
2. Select the System tab, containing **FF00** information.
3. Configure the properties specifies below in Table 2-2:

Table 2-2: Full Administrator Configuration Attributes

Attribute	Valid Range	Notes
<b>(TP) - Token Passing Type</b>	0 = Irresponsible Peer 1 = Full Administrator	dictates the network mode of the GPC.
<b>(U1) through (UC) - Peer Units</b>	0-32768	define PUP peers that the GPC will pass a network token to in the event it receives a network token from a Full Administrator, or is configured as a Full Administrator.

## 2.4 TIME AND DATE CONFIGURATION

All Time and Date configuration is stored under the System>Time/Date area.

All GPC controllers include a real-time clock. The real-time clock is used to allow the GPC to perform local scheduling through use of Schedule channels, as well as permits the GPC to be a time-master to a local PUP network.

By default, the time and date of the GPC are calibrated for eastern standard time (EST). It may be necessary for you to configure the time and date to match your locale. The current time is controlled through the **(TM) Current Time** attribute. In BACnet, time is reported in military format (24 hour). The current date is controlled through the **(DT) Current Date** attribute.

To change either attribute, you can simply write the correct values to these properties. Alternatively, if you have multiple GPC controllers, you may perform a Time Synchronization Broadcast using SoloPro, allowing you to send the current time and date of your PC, or a time and date you define. Upon receiving the synchronization command, the GPC controllers will automatically update the **(TM) Current Time** and **(DT) Current Date** attributes.

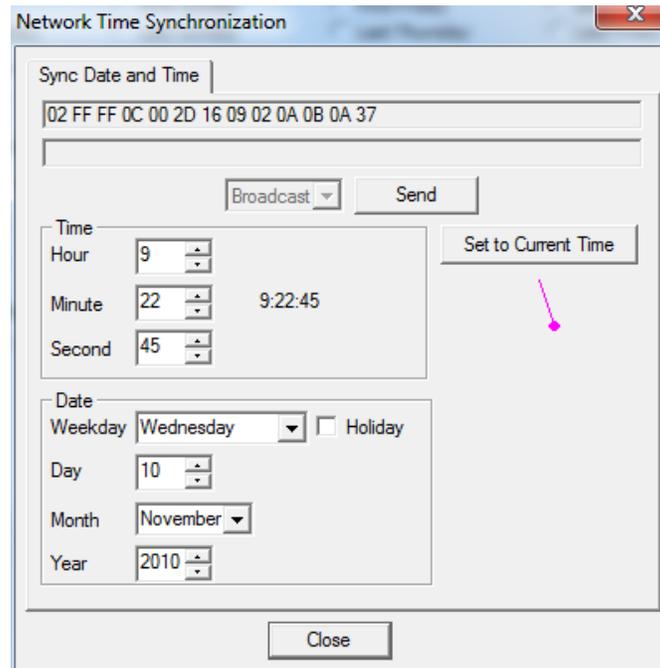


Figure 2-1 - Time Sync via SoloPro

### 2.4.1 TIME SYNCHRONIZATION

By nature, all SBC-GPC controllers are capable of receiving and interpreting time-synchronization messages from other PUP devices.

More so, SBC-GPC controllers can be configured to transmit time-synchronization messages to the local PUP network it is connected to; effectively making it a network time master. This may be necessary if you have a network of lower-level devices that do not contain real-time clocks by default (such as SBC-VAV, SBC-ASC, or SBC-SD devices).

Setup for this function is located under the System>Time/Date area your SBC-GPC when monitoring via SoloPro.

Programming the GPC to perform this function consists of configuring two attributes:

- **(BS) Broadcast Sync Time** - enables or disables the device to send time broadcast messages.
- **(BT) Broadcast Time Synch Interval** - defines how often, in minutes, the GPC sends transmits a time-synchronization message to the network.

CAUTION	
	<p>Your GPC must be configured as a Full Admin, or be passed a network token from a Full Administrator if you wish for it to be the dedicated time master.</p> <p>For optimal results, AAM recommends one time master per local EIA-485 network.</p>

Once configured, all devices on the local network will receive time broadcasts and process them.

#### 2.4.1.1 CONFIGURING THE BROADCAST TIME SYNC INTERVAL

The frequency of how often time synchronization messages are sent to the network is controlled through the **(BT) Broadcast Time Synch Interval** attribute. This attribute specifies how often, in minutes, the GPC will send a time-synchronization to the network

Table 2-3: Broadcast Sync Time

attribute	Valid Range	Notes
<b>(BS) Broadcast Sync Time</b>	0 or 1 (No or Yes)	A value of 0 disables time-synchronization message transmissions.

#### 2.4.2 DAYLIGHT SAVING

The SBC-GPC can be programmed to automatically update its clock based on daylight saving time. There are several properties in the GPC that correspond to daylight saving, and they can be configured to meet rules in your regional location.

The **(DS) Daylight Savings Status** attribute defines if the controller is currently in daylight savings mode.

**(RD) Daylight Saving Start Day**, **(RM) Daylight Saving Start Month**, and **(ST) Daylight Saving Start Time** control when daylight saving starts in your regional location. Several day options are provided based on universal daylight saving schedules, as well as each month of the year, and a configurable time.

**(ND) Daylight Saving End Day**, **(NM) Daylight Saving End Month**, and **(ET) Daylight Saving End Time** control when daylight savings ends in your regional location. Similar to the starting properties, the same options are provided for defining end parameters.

Once these parameters are configured, the GPC will automatically track when to begin and end daylight saving based on its real-time clock.

**CAUTION**



It is important to make sure that the **(CT) Current Time** and **(DT) Current Date** attributes are correctly configured or synchronized when utilizing the Daylight Savings features.

---

## SECTION 3: INPUTS SETUP

*This section describes the process of setup and configuration for hardware inputs - reviewing Universal Inputs (which can be Analog or Binary) and Digital Inputs (optically isolated, high-speed pulse-counting inputs), their software characteristics, as well as the setup of Piecewise Curves for custom Analog Input sensor configurations*

### IN THIS SECTION

Inputs Overview .....	3-3
Universal Input Overview .....	3-3
Digital Inputs Overview .....	3-3
Programming Concepts and Techniques .....	3-3
Universal Inputs .....	3-5
IVR Jumpers for Universal Inputs .....	3-5
Connecting Universal Inputs .....	3-6
Universal Input Configuration .....	3-7
Configuring Analog Input Alarm/Event Notifications .....	3-11
Using Universal Inputs for Pulse Counting Applications .....	3-12
Configuring Digital Input Alarm/Event Notifications.....	3-13
Digital Inputs .....	3-14
Connecting Digital Inputs .....	3-14
Configuring the Digital Inputs .....	3-15
Piecewise Curves .....	3-16
Piecewise Curves for Voltage Inputs.....	3-16
Piecewise Curves for Current Inputs.....	3-18
Piecewise Curves for Resistance Inputs.....	3-18



## 3.1 INPUTS OVERVIEW

SBC-GPC product models support Universal Inputs, capable of being configured to monitor analog and binary signals. Models of GPC products support either on-board I/O or expansion modules via STATbus. While the total amount of inputs per GPC will depend on the model number, setup and configuration of outputs is exactly the same across the product family. The table below provides information on-board I/O support, as well as expansion support.

### 3.1.1 UNIVERSAL INPUT OVERVIEW

Universal Inputs are jumper configured to sense voltage, resistance, current, and other signal forms as reviewed in this section.

*Table 3-1: GPC Controller Models and Analog Inputs*

Controller Model	On-Board	Expandable
SBC-GPC1	12 on-board	12 additional
SBC-GPC2	8 on-board	4 (reserved for STAT/RHT sensors)
SBC-GPC3	n/a	24 additional

The software configuration of Universal Inputs also provides options for software voltage and current scaling, support for alarm/event notifications, and low-speed pulse counting.

### 3.1.2 DIGITAL INPUTS OVERVIEW

Digital Inputs are different from standard Universal Inputs. Digital Input products are designed to perform high-speed pulse counting, as well as input monitoring on SSB-DOx devices that contain inputs. Using various STATbus IOX modules, you may add additional Digital Inputs to a controller if desired.

*Table 3-2: GPC Controller Models and Digital Inputs*

Controller Model	On-Board	Expandable Digital Inputs
SBC-GPC1	1 on-board	7 additional
SBC-GPC2	1 on-board	n/a
SBC-GPC3	n/a	8 additional

The software configuration of Digital Inputs also provides support for alarm/event notifications.

### 3.1.3 PROGRAMMING CONCEPTS AND TECHNIQUES

To enhance your programming experience, the following are a few helpful concepts and techniques to keep in mind when using these objects.

#### 3.1.3.1 MAKE THE (ON) INPUT NAME UNIQUE

The GPC supports the ability to allow each input's name to be assigned a custom value. By default, the software uses generic names for inputs. For ease of programming and flow, it is strongly recommended

that you change the **(ON) Input Name** of any used Input channels. This allows you not only to keep better track of which channels have been used, but also allows you to easily troubleshoot your linked logic.

#### 3.1.3.2 ENABLE ALARMING WHEN NEEDED

All Input objects optionally support alarming. When alarming is disabled **(AE) Alarm Enable = Disabled (0)**, alarming may be configured at the AspectFT level if desired, or can be done at the controller level when **(AE) Alarm Enable** is configured to a non-Disabled value.

## 3.2 UNIVERSAL INPUTS

Each universal input may operate as either a digital or analog input. Each input may be configured individually to read any of the following:

- digital (on/off)
- linear inputs (0-5 V, 0-10 V, 0-20 mA, 4-20 mA, etc.) scaled between a programmable minimum and maximum value
- non-linear inputs with response provided via programmable Piecewise Curve objects
- thermistor (Precon type III 77° @ 10kΩ)
- low-speed pulsing
- SMARTStat device (using available instances higher than the pre-assigned on-board)

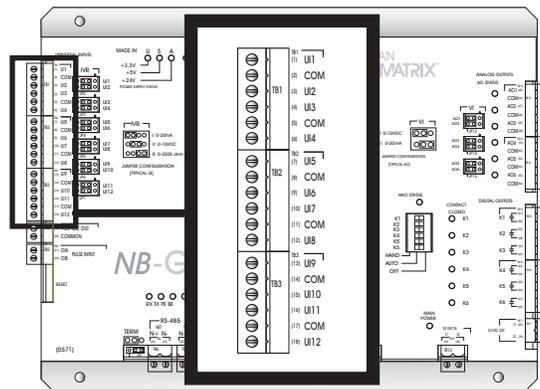


Figure 3-1: Example Location of the Universal Inputs on a GPC1

### 3.2.1 IVR JUMPERS FOR UNIVERSAL INPUTS

The IVR jumpers are located next to terminal block TB2 as shown in Figure 3-2.

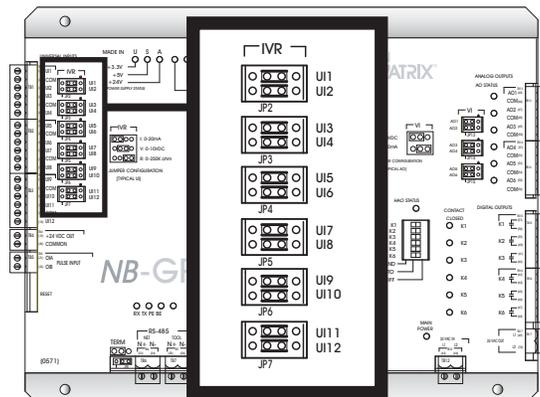


Figure 3-2: Location of the IVR Jumpers for Universal Inputs

The IVR jumpers are used to configure GPC hardware for the different types of signals that can be connected to the associated universal input point. By moving the jumper to different positions the associated point can be selected as either a current, voltage, or resistance input. The possible positions for the IVR jumper are shown in Figure 3-3.

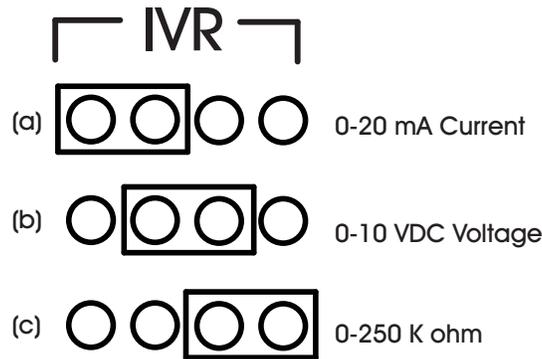


Figure 3-3: IVR Jumper Positions

When the jumper is in the left-most position (the “I” position), shown in Figure 3-3a, the point is configured as a 0-20 mA current sensor. When the jumper is in the center position (the “V” position), shown in Figure 3-3b, the point is configured as a 0-10 V voltage sensor. When the jumper is in the right-most position (the “R” position), shown in Figure 3-3c, the point is configured as a 0-250 kΩ resistive sensor. The “R” position would be used for thermistor inputs as well as dry contact, digital inputs. The default position for the IVR jumper is the “R” position.

**NOTE**



The IVR jumpers for unused Universal Inputs should be left in (or, if the jumper has been moved, returned to) the default position, “R”.

**CAUTION**



*Leaving the IVR jumpers of an unused Universal Input in the “V” position can lead to inaccuracies in other Universal Inputs. A similar situation can exist if the jumper is not connected to any terminals, for example, if the jumper had been lost.*

### 3.2.2 CONNECTING UNIVERSAL INPUTS

Devices are connected to the on-board universal input points on the GPC via terminal blocks located on the left side of the controller. Each connector has points for universal inputs and commons. There are fewer common points than universal input points, so adjacent points must share common terminals. For example on an GPC1, UI5 and UI6, located at terminal 7 and 9 respectively, would both be connected using the common ground on terminal 8.

Input devices are connected to a universal input point by connecting one of the wires to the associated terminal, labeled UI1 through UI12, and the other to the adjacent common terminal, labeled COM.

## CAUTION



*I/O Common (COM terminals) on GPC products are shared and are not at the same potential as the chassis of the controller. I/O Common can be connected to chassis or panel ground, provided that the ground connection is known to be good.*

## 3.2.3 UNIVERSAL INPUT CONFIGURATION

The following sub-sections discuss the configuration of universal inputs.

## 3.2.3.1 LOOP POWERED UNIVERSAL INPUTS

For inputs that require power, the GPC products have a 24 VDC power connection capable of providing up to 150 mA via terminals 19 and 20 on terminal block TB4. This is enough current to power approximately 6-7 loop powered input devices. An example of a loop powered input connected to an GPC1 is shown in Figure 3-4.

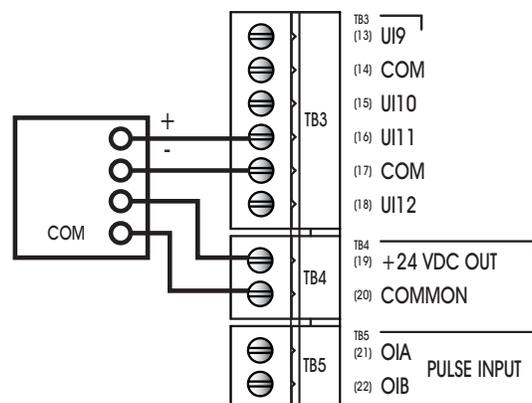


Figure 3-4: Using Loop Powered Universal Inputs

## CAUTION



*The 24VDC power output provides 150 mA power output. If the number of end devices require more power than provided, use an external transformer.*

## 3.2.3.2 CURRENT INPUTS

Any sensor which generates a signal in the form of a current is classified as a current sensor. Ranges of 0-20 mA and 4-20 mA are common in sensors. The current produced by these sensors is often proportional to the value being measured. For example, if a pressure sensor measured 0 to 5 inches of water gauge and had an output range of 0 to 20 mA, then a reading of 10 mA would correspond to a pressure of 2.5 inches of water gauge.

The first thing that needs to be done so that a universal input can be used as a current input is to make sure that the IVR jumper is in the correct position (See “IVR Jumpers for Universal Inputs”, Section 3.2.1). In this case, you would make sure that the IVR jumper for the input was set to the “I” position.

Once the type of sensor has been set, you need to use SoloPro. Configuration entails telling the GPC what type of sensor is connected to an input and then specifying the range of values over which that sensors operates.

To specify the sensor type, you must navigate to the tab for the corresponding the Universal Input you are configuring. Next, you want to set the **(ST) Sensor Type** attribute to a value of 3 for 4-20 mA inputs, or 8 for 0-20 mA inputs. That tells the GPC that a current sensor is connected.

You also need to specify the range of values in which the sensor operates. This is necessary so that the GPC can calculate the measured value from the input signal. The **(MN) Minimum Scaled Value** attribute should be set to the lowest value that your sensor can measure. The **(MX) Maximum Scaled Value** attribute should be set to the maximum scaled value for your input.

As an example, sensors which measure relative humidity are often current sensors that operate in the 4-20 mA range. For a sensor of this type you would set **(ST) Sensor Type=3** because the sensor measures 4-20 mA. Relative humidity ranges from 0 to 100% so you would set **(MN) Minimum Scaled Value=0** and **(MX) Maximum Scaled Value=100** to represent the limits of the sensor’s output. In this case, a raw value of 4 mA would be scaled to a value of 0% in engineering units. The relative humidity sensor would read 100% if the input were reading a signal of 20mA. Figure 3-5 shows a typical 2-wire current sensor connected to the GPC. Figure 3-5a shows the sensor using an external power supply and Figure 3-5b shows the same sensor using the GPC’s on-board 24 VDC power output to power the sensor.

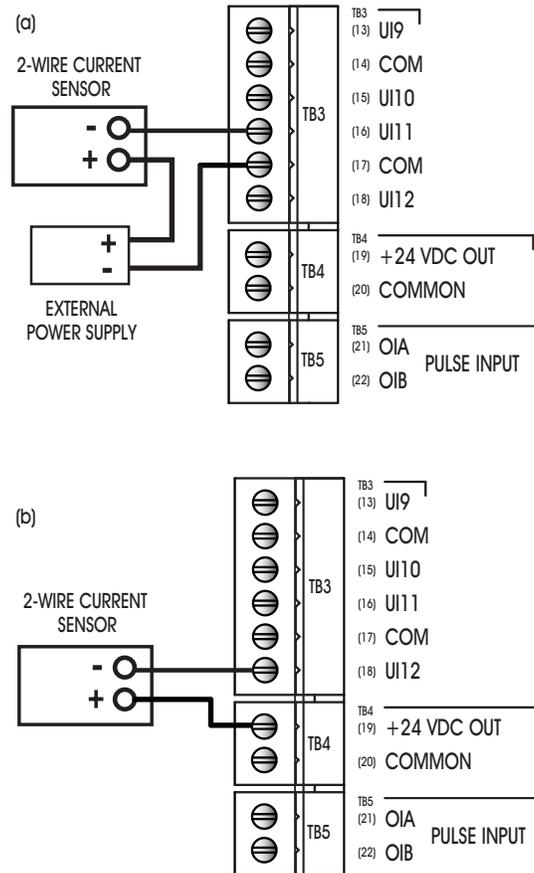


Figure 3-5: Connecting a current sensor

### 3.2.3.3 VOLTAGE INPUTS

Any sensor which puts out a voltage in response to a measured value is classified as a voltage sensor. Voltage sensors behave in very much the same way as current sensors. The primary differences have to do with the internal circuitry of the GPC and how the signal is read.

The first thing that needs to be done so that a universal input can be used as a voltage input is to make sure that the IVR jumper is in the correct position (See "IVR Jumpers for Universal Inputs", Section 3.2.1). In this case, you would make sure that the IVR jumper for the input was set to the "V" position.

Second, you must tell the GPC what type of sensor is connected to an input and then specifying the range of values over which that sensors operates.

To specify the sensor type, you must to go to Universal Input object corresponding to the input you are configuring. Next, you want to set the **(ST) Sensor Type** attribute to a value of 2, for 0-5 V inputs, or 6, for 0-10 V inputs.

NOTE	
	<p>If a voltage input is unplugged from the controller, the IVR jumper configuration should be set to R until the voltage input is reconnected.</p>

You also need to specify the range over which the sensor operates. This is necessary so that the GPC can calculate the measured value from the input signal. The **(MN) Minimum Scaled Value** attribute should be set to the lowest value that your sensor can measure. This would correspond to the reading at zero volts. The **(MX) Maximum Scaled Value** attribute should be set to the maximum scaled value for your input. For example, if a 0-10 V carbon dioxide sensor measuring from 0-5000 ppm would have **(MN) Minimum Scaled Value** would be set to 0 and **(MX) Maximum Scaled Value** would be set to 5000.

3.2.3.4 THERMISTOR INPUTS

The thermistor is one of the most common types of resistive sensors for temperature measurement. The thermistor’s combination of high accuracy over a wide range coupled with its low cost makes it one of the most popular temperature sensors used. Because of the thermistor’s popularity, the GPC has the response curve for a **precon type III** thermistor built in.

Assume that the thermistor you wish to configure is connected to UI6 as shown in Figure 3-6. The first thing that needs to be done to configure a universal input to be used as a thermistor input is to make sure that the IVR jumper is in the correct position (See “IVR Jumpers for Universal Inputs”, Section 3.2.1). In this example, you would make sure that the IVR jumper for UI6 was set to the “R” position.

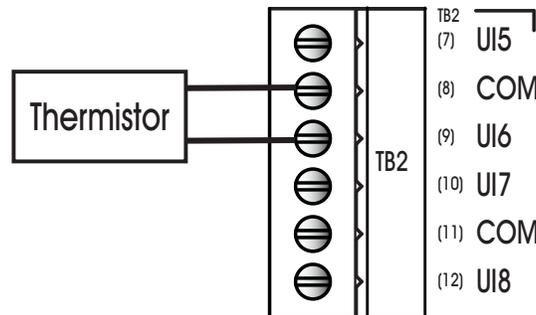


Figure 3-6: Connecting a Thermistor

Once the jumper is properly set, the second thing that must be done is to set the sensor type. This tells the GPC what kind of sensor is connected to the input. Communicating with the controller via SoloPro, you will navigate to the corresponding Universal Input. Set the **(ST) Sensor Type** attribute equal to 7, the value which corresponds to a thermistor. The GPC will then automatically set the **(MN) Minimum Scaled Value** to -35.0 and the **(MX) Maximum Scaled Value** to 240.0, the minimum and maximum values that can be read by this type of sensor. The temperature will now be displayed in the **(CV) Current Value** attribute of this channel. The value of **(CV) Current Value** will also be displayed in the Universal Input Summary channel.

At this point, it is helpful to give the channel a name so it can be easily identified in the future. You can name the channel by setting the value of the **(ON) Input Name** attribute. You should then check the reading and adjust any disagreement between the sensor and a known reading using the **(OF) Input**

**Offset** attribute, This specifies a fixed offset for the sensor and would be used if the sensor reading was off by a fixed amount. For example, if a sensor was reading three degrees below the actual room temperature. In that case, you would set **(OF) Input Offset=3.0** to correct the reading.

#### 3.2.3.5 NON-LINEAR INPUTS

The thermistors discussed previously are just one example of a sensor whose output characteristics are well defined. Because Precon type III thermistors are so prevalent, the output response curve is included with the controller. However, there are a number of other common sensors whose response is non-linear. For these types of inputs, the GPC provides the option of using one of the eight available Piecewise Curve channels to specify the response of the input device.

The Piecewise Curves can be used with current, voltage, or resistive inputs. Before configuring the Piecewise Curve, make sure that the IVR jumper is in the correct position for the type of sensor being used. See "IVR Jumpers for Universal Inputs", Section 3.2.1 for more information on setting the IVR Jumper.

To use an input with a non-linear response, you must define the Piecewise Curve and then set the input to use the curve you have defined to scale its readings. The process for defining the Piecewise Curve is given later in this manual. The response data necessary to construct the curve will usually be available in the catalog from which the sensor was ordered or on the data sheet accompanying the sensor.

To set the input to use the Piecewise Curve, set the **(ST) Sensor Type** attribute for the input equal to one of the available Piecewise Curve channels. This tells the GPC to use the specified Piecewise Curve.

At this point, you should give the channel a name so that you can easily identify it in the future. You can name the channel by writing to the **(ON) Input Name** attribute.

#### 3.2.4 CONFIGURING ANALOG INPUT ALARM/EVENT NOTIFICATIONS

Analog Inputs can be configured to support alarm/event notifications. To enable alarming for Analog Inputs, set **(AE) Enable Alarming** for a valid alarm application.

Analog-based inputs objects can be configured to trigger one of the two following conditions:

- . Low Limit - occurs when **(CV) Current Value** is less than the value specified in **(LL) Low Limit**.
- . High Limit - occurs when **(CV) Current Value** is greater than the value specified in **(HL) High Limit**.

#### 3.2.5 DIGITAL INPUT CONFIGURATION

An input that only has two signal states is considered a digital input. The most basic digital inputs are switches or contacts. The switch is either on or off, the contact is closed or open. Inputs of this type have many uses in a building automation system.

Despite only having two possible states, digital inputs require a bit more configuration than their universal counterparts. Like a universal input, you should first check to make sure that the IVR jumper is in the correct position (See "IVR Jumpers for Universal Inputs", Section 3.2.1). For a digital input, you want to set the jumper to the "R" position. This is used because an open contact would have a very high resistance while a closed contact would have a very low resistance, making it easier to detect the states.

For a digital sensor, you will set the **(ST) Sensor Type** attribute to "Digital" (0).

Digital inputs can take one of two states, but you can tell the controller how you want it to treat those states. By setting the **(IP) Input Polarity** attribute you can specify whether the GPC should display **(CV) Current Value =1** for a high signal (normal operation, **(IP) Input Polarity=0**) or a low signal (reverse operation, **(IP) Input Polarity=1**).

Digital inputs also have a **(RH) Run Hours** attribute. This attribute tracks the amount of time (in hours) that the sensed signal has been active.

### 3.2.6 USING UNIVERSAL INPUTS FOR PULSE COUNTING APPLICATIONS

Digital Inputs can be used to perform pulse count applications. Both Universal Inputs, as well as opto-isolated Digital Inputs can be used in this application. Each has limitations and advantages over the other:

- Universal Inputs can detect pulses at a maximum of 10hz. No more than three universal inputs per controller should be used at a time to perform pulse counting. Universal Inputs configured as such must have accurate thresholds configured in order for valid pulses to be detected.
- Digital Inputs (opto-isolated) can detect pulses at a maximum of 30hz. In order for a positive signal to be detected, the signal must be at least 3vdc or greater.

Universal Inputs configured as digital inputs can be used to perform low-speed pulse counting for applications that require less than 10Hz accuracy. To configure the digital input to perform pulse counting, you must first specify the corresponding input's IVR jumper setting via **(VR) IVR Setting (For Pulse Counting Mode)**. To ensure counting accuracy, this setting must match the jumper configuration. Then, a **(PT) Pulse Threshold**, which controls must also be specified, which indicates the minimum voltage, amperage, or resistance that the Universal Input must "sense" in order for a pulse to be detected.

#### CAUTION



*IOX Modules that contain Universal Inputs such as the SSB-UI1 do not support pulse counting applications. If you need additional Pulse Counting inputs, you must use the SSB-DI module, or use an SSB-IOX module device that contains an opto-isolated Digital Input.*

#### NOTE



In optimal applications, no more than three (3) Universal Inputs should be configured to perform pulse counting accurately at 10Hz.

To configure the pulse counting application, use **(MD) Pulse Counting Mode** to specify how pulse inputs are counted. Pulses can be sensed using Rising Edges, Falling Edges, or Both. Then, configure **(SF) Scaling Factor** to apply against the raw count (displayed via **(NP) Number of Pulses**) to determine the ultimate **(SV) Scaled Value**.

**CAUTION**

*GPC products manufactured after April 2010 can have their Universal Inputs configured to perform pulse counting. Devices manufactured before this date cannot perform pulse counting via Universal Inputs.*

### 3.2.7 CONFIGURING DIGITAL INPUT ALARM/EVENT NOTIFICATIONS

Digital Inputs can be configured to support alarm/event notifications. To enable alarming for an Binary Inputs, set **(AE) Alarm Enable** for a valid alarming application. Once configured, additional properties will become available that control the setup and configuration of how alarms/events are handled by the object.

### 3.3 DIGITAL INPUTS

Some GPC controller models include an optically isolated digital input with dedicated pulse counting features. These digital inputs are capable of detecting signals in the range 3-40 VDC peak to peak or 2-29 VAC at 50/60 Hz. There is one on-board, digital input point on the GPC1, and GPC2 mapped to Digital Input 1, and seven additional objects, Digital Inputs 2-8, provided so that additional pulse counting, digital input points may be added via STATbus devices. An example of the GPC1 on-board digital input is located at terminals 21 and 22 on TB5 as shown in Figure 3-7.

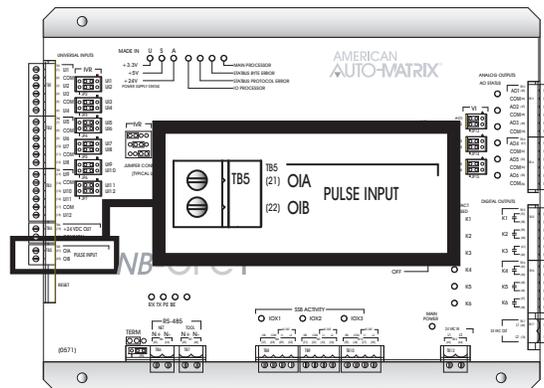


Figure 3-7: Location of the Digital Input

The digital inputs operate at a much higher frequency than a Universal Input. This makes it possible for the input to not only detect whether a signal is on or off, but to detect rapid pulses from the input. These pulses would be used primarily for demand metering applications as part of an energy management system. Digital, pulse counting inputs can also be used in flow metering applications.

Regardless of the specific application, all pulse counting digital devices operate on the same principle. The device will generate a pulse for a given quantity of the value that is being measured for a demand metering application. One pulse might correspond to one kilowatt-hour of power whereas, for a flow metering setup, a single pulse might correspond to one gallon of liquid. The important piece of information is the correlation between pulses and measured values.

#### 3.3.1 CONNECTING DIGITAL INPUTS

The on-board digital input on the GPC is a wet contact connected to the OIA and OIB terminals, terminals 21 and 22 on the GPC, via terminal block TB5. The proper way to connect the input is shown in Figure 3-8.

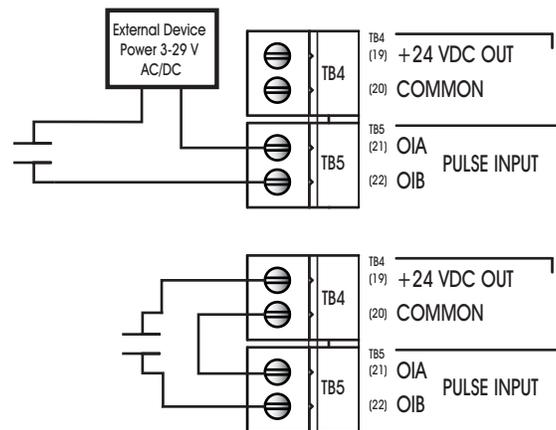


Figure 3-8: Connecting a sensor using external power (top) and the GPC power output (bottom) to the Digital Input

### 3.3.2 CONFIGURING THE DIGITAL INPUTS

To configure a digital pulse counting input, you must set the **(MD) Pulse Counter Mode** attribute to “Rising Edge”, “Falling Edge”, or “Both”. This tells the GPC that you want the input to operate as pulse counter and not a simple digital input.

To correlate pulses with the value being measured, you must enter a value for the **(SF) Scaled Factor** attribute. This multiplier specifies the amount of the measured quantity that is accumulated for each pulse. For example, if a demand meter sends out a pulse for each kilowatt-hour of power used, then you would set **(SF) Scaled Factor** = 1.0 since one pulse corresponds to one kilowatt-hour. If your input device were a flow meter that sent a pulse for every ten gallons of liquid that passed through a pipe, then you would set **(SF) Scaled Factor** =10.0.

The total number of pulses accumulated will be displayed in the **(NP) Number of Pulses Accumulated** attribute. While this can be useful, you will be more concerned with the scaled pulse count stored in the **(SV) Scaled Value** attribute. This is the value of the total number of pulses, **(NP) Number of Pulses Accumulated**, multiplied by the scaling factor, **(SF) Scaled Factor**. **(SV) Scaled Value** gives you the total amount of the value that you are measuring. For example, if the GPC has accumulated 1250 (**(NP) Number of Pulses Accumulated** =1250) pulses and each pulse corresponds to 2.5 gallons of liquid that has been pumped through a pipe (**(SF) Scaled Factor**=2.5), then the total amount of liquid pumped (**Neptis**) would be displayed in **(SV) Scaled Value**. In this case, **(SV) Scaled Value** would have a value of 3125, meaning that 3125 gallons of liquid had been measured.

## 3.4 PIECEWISE CURVES

GPC products can accommodate non-linear sensors by using built-in tables to define the response characteristics of an Analog Input sensor. Each table requires eleven points to define ten linear segments which approximate the response of the sensor. The controller will perform a linear interpolation to 'look up' values that lie along an individual segment much like the calculations performed by the Scale objects.

### 3.4.1 OVERVIEW OF X AND Y

Attributes **(X1) Point 1's value in % Full Scale** through **(XB) Point 11's value in % Full Scale** represent the sensor readings for eleven chosen points on a sensor curve. The acceptable range for **X1** through **XB** depend on the chosen sensor type. For a voltage input, the 0-10 V input range is mapped to **X1** through **XB** values from 0 through 100. A current input, with a range of 0-20 mA, can have **X1** through **XB** values from 0 through 50. The 0-250 k $\Omega$  range for a resistive input can have **X1** through **XB** values from 0 through 25. The values of **X1** through **XB** must be entered in increasing order (i.e. **X1** < **X2** < **X3** etc.).

#### NOTE



The Piecewise Curve will only interpolate values for input values between **X1** and **XB**. If the input is below **X1**, the Piecewise Curve will be pegged at the value associated with **X1**. If the input is above **XB**, the Piecewise Curve will be pegged at the value associated with **X1**.

Attributes **(Y1) Point 1's value in engineering units** through **(YB) Point 11's value in engineering units** are the Engineering Unit values (e.g., 70 degrees, 72 degrees, etc.), corresponding to the sensor readings entered into **X1** through **XB**. These values, coupled with the corresponding sensor readings, define the line segments which make up the piecewise curve.

### 3.4.2 PIECEWISE CURVES FOR VOLTAGE INPUTS

To program a piecewise curve for a nonlinear sensor, you need to know the response characteristics of the sensor. These response characteristics are usually supplied by the manufacturer and may be in the form of a graph or table. Figure 3-9 shows an example of what a curve for a temperature sensor may look like. Though the sensor response could extend beyond this range, you will get more accurate results if you limit the range of your Piecewise Curve to the range of values you expect to see from the sensor. Figure 3-9 only contains the portion of the sensor's response that would be needed for zone temperature monitoring.

Once you have the response data, in either graph or table form, you must choose the points which define the line segments that approximate the response curve in the expected response region. When choosing the points to use, you can use fewer points in areas of the curve that are mostly linear and concentrate the points to better approximate the more non-linear portions of the response. In Figure 3-9, you can see that more points are chosen near the 'bends' in the characteristic response curve.

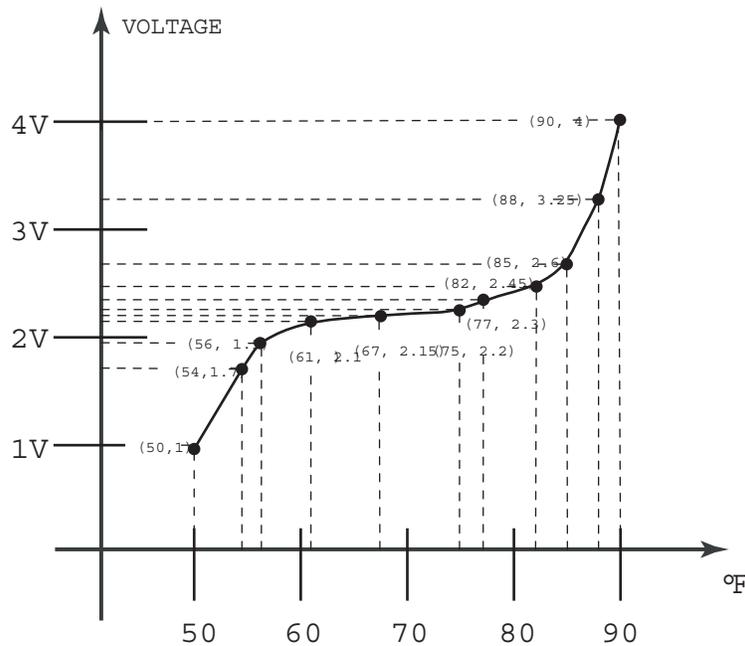


Figure 3-9: An Example of a Sensor Response Curve

From the graph, the following values are selected to represent the curve:

Table 3-3 : Sensor Response Points

Voltage	Temperature (°F)
1.00	50
1.70	54
1.90	56
2.10	61
2.15	67
2.20	75
2.30	77
2.45	82
2.60	85
3.25	88
4.00	90

Next you must convert the voltage values into a percentage of full scale to be used to define the x-coordinates of your piecewise curve. Since the voltage input has a range of 0-10 V, each volt measured corresponds to ten percent of the full scale. The formula for the percentage of full scale output for a voltage sensor is simply:

$$\% \text{ Full Scale} = \text{Voltage} \times 10$$

The calculated percentages correspond to **X1** through **XB** and the temperature reading correspond to **Y1** through **YB**. For the sensor described above, this gives the following assignments:

*Table 3-4 : Assigning Sensor Response Points to the Piecewise Curve*

	% Full Scale	Temperature (°F)	
<b>X1</b> ⇐	10.0	50	⇒ <b>Y1</b>
<b>X2</b> ⇐	17.0	54	⇒ <b>Y2</b>
<b>X3</b> ⇐	19.0	56	⇒ <b>Y3</b>
<b>X4</b> ⇐	21.0	61	⇒ <b>Y4</b>
<b>X5</b> ⇐	21.5	67	⇒ <b>Y5</b>
<b>X6</b> ⇐	22.0	75	⇒ <b>Y6</b>
<b>X7</b> ⇐	23.0	77	⇒ <b>Y7</b>
<b>X8</b> ⇐	24.5	82	⇒ <b>Y8</b>
<b>X9</b> ⇐	26.0	85	⇒ <b>Y9</b>
<b>XA</b> ⇐	32.5	88	⇒ <b>YA</b>
<b>XB</b> ⇐	40.0	90	⇒ <b>YB</b>

### 3.4.3 PIECEWISE CURVES FOR CURRENT INPUTS

Where the full scale of the voltage sensor is represented internally as a full 0 to 100%, a current sensor is represented in slightly less than 50% of the full scale readable by the controller. This means that the 0 to 20 mA full scale sensor reading range is mapped to the range of 0 to slightly less than 50% of the full scale readable by the controller. Calculating the Piecewise Curve for a current input is the same as for the resistive input, except that you would instead apply a different formula to calculate the percentage of full scale. Here, 1 mA read in from the sensor corresponds to 2.49% of the full scale. You can simply multiply the current value from the sensor’s characteristic response, or you can use the following formula to calculate the percentage of full scale:

$$\% \text{ Full Scale} = \frac{\text{Current(mA)} \times 249}{100}$$

The values for **Y1** through **YB** are entered in Engineering Units in exactly the same way as for the voltage sensor.

### 3.4.4 PIECEWISE CURVES FOR RESISTANCE INPUTS

Like the current sensor, the resistance sensor is represented inside the controller a fraction of the full scale range possible in the controller. The 0 to 250 kΩ resistance range is represented internally as 0 to slightly less than 25% of the full scale readable by the controller. Calculating the a Piecewise Curve for a resistive

input is also slightly different than for the voltage or current sensors because the controller measures the voltage drop across the input and that response is inherently non-linear. Because of this, there is no simple multiplier that can be used to convert resistance to full scale percentage as for the voltage or current sensors. Instead, you will have to use the following equation:

$$\% \text{ Full Scale} = 25 \times \frac{\text{Resistance}(\Omega)}{\text{Resistance}(\Omega) + 20000}$$

The calculated full scale values are then entered into **X1** through **XB**. The values for **Y1** through **YB** are entered in Engineering Units in exactly the same way as for the voltage and current sensors.



---

## SECTION 4: OUTPUTS SETUP

*This section describes the process of setup and configuration for hardware outputs - including Analog Outputs and Digital Outputs and their software characteristics.*

### IN THIS SECTION

Outputs Overview.....	4-3
Analog Output Overview .....	4-3
Digital Output Overview .....	4-3
Programming Concepts and Techniques .....	4-3
Analog Output Summary.....	4-4
Analog Outputs .....	4-5
Configuring Minimum and Maximum Thresholds.....	4-5
Control Mode .....	4-5
Input Interlocking.....	4-5
Additional Interlock.....	4-5
Actual Output Voltage and Actual Output Current.....	4-5
Communication Failure Enable Positioning .....	4-6
Fire Override .....	4-6
Update Threshold .....	4-6
Run Hours .....	4-6
Digital Output Summary .....	4-7
Digital Outputs.....	4-9
Configuring Minimum Off/On Times .....	4-9
Configuring Polarity.....	4-9
Actual Output State .....	4-9
Control Mode .....	4-9
Input Interlocking.....	4-9
Additional Interlock.....	4-9
Schedules to Follow .....	4-10
Inrush Current Lockout .....	4-10
Pulse Width.....	4-10
Run Hours .....	4-10



## 4.1 OUTPUTS OVERVIEW

GPC product models support both Analog and Digital Outputs for direct equipment control using either on-board I/O or expansion modules via STATbus. While the total amount of outputs per GPC will depend on the model number, the setup and configuration of outputs is exactly the same across the product family. The table below provides information on-board I/O support, as well as expansion support.

### 4.1.1 ANALOG OUTPUT OVERVIEW

Analog Outputs have jumpers configured to output voltage in either 0-10vdc, or 0-20mA.

*Table 4-1: GPC Controller Models and Analog Outputs*

Controller Model	On-Board	Expandable Outputs
SBC-GPC1	6 on-board	6 additional
SBC-GPC2	4 on-board	n/a
SBC-GPC3	n/a	12 additional

The software configuration of Analog Outputs also provides options for software voltage and current scaling.

### 4.1.2 DIGITAL OUTPUT OVERVIEW

Digital Outputs on the GPC controller itself are triac outputs. Using various STATbus IOX modules, you may add either triac or relay digital outputs.

*Table 4-2: GPC Controller Models and Binary Outputs*

Controller Model	On-Board	Expandable Outputs
SBC-GPC1	6 on-board	6 additional
SBC-GPC2	5 on-board	n/a
SBC-GPC3	n/a	12 additional

The software configuration of Digital Outputs also provides options for minimum on and off timing, and output staging.

### 4.1.3 PROGRAMMING CONCEPTS AND TECHNIQUES

#### 4.1.3.1 MAKE THE (ON) CHANNEL NAME UNIQUE

The GPC supports the ability to allow each channel's name to be assigned a custom value. By default, the software uses generic names for channels. For ease of programming and flow, it is strongly recommended that you change the channel name of any used channel. This allows you not only to keep track of which channel have been used, but also allows you to easily troubleshoot your linked logic.

## 4.2 ANALOG OUTPUT SUMMARY

The Analog Output Summary object provides a summary overview of all Analog Outputs on the GPC. Through this channel, a user can easily view key information regarding all of the Analog Outputs. Information that can be viewed from this area includes:

- . Current Values - indicates the **(CV) Current Value** for each Analog Output.
- . Outputs in Automatic Mode - indicates the current state of the **(AM) Override** attribute for each Analog Output object.
- . Outputs with Unreliable Values - indicates, in bitmap form, which Analog Output objects are unreliable. An object could be unreliable based on value scaling, actual output status, or a fail based on STATbus communication (for IOX modules), or because it has not been assigned to a STATbus module.
- . Outputs Enabled for Comm Failure - indicates outputs that have been enabled for communication failure positioning.

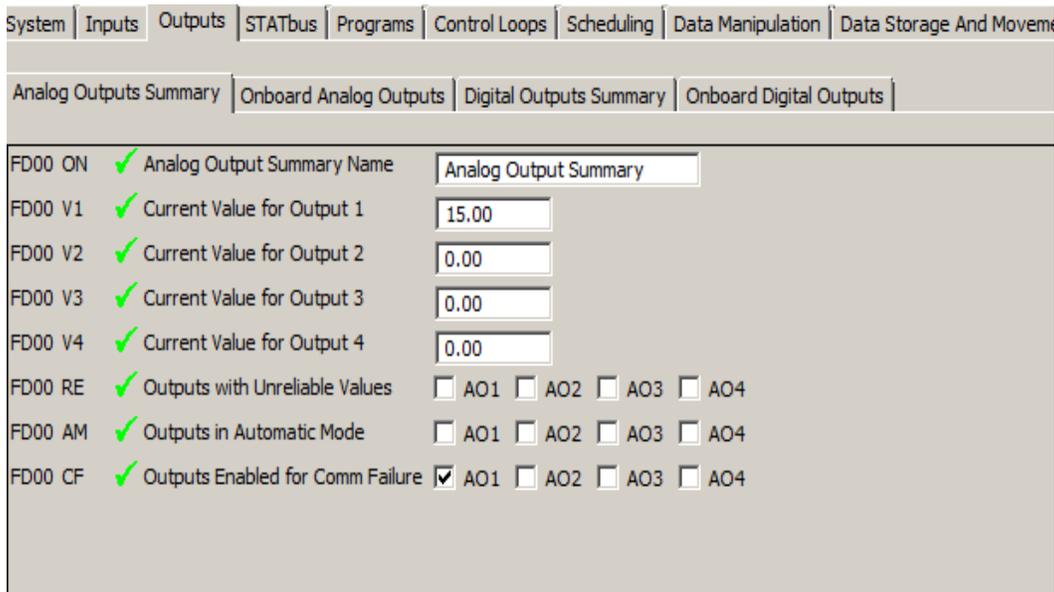


Figure 4-1 Analog Output Summary

## 4.3 ANALOG OUTPUTS

Analog Outputs are used to provide proportional control signals in either 0-10vdc or 0-20mA output form.

### 4.3.1 CONFIGURING MINIMUM AND MAXIMUM THRESHOLDS

Analog Outputs can be software configured to accept a minimum or maximum value. Additionally, the output itself can be mathematically scaled to establish a percentage of the maximum voltage or current.

The **(LS) Minimum Scaled Output** and **(HS) Maximum Scaled Output** attributes specify the minimum and maximum scaled values for the outputs, expressed as a percentage of the full scale. **(MN) Minimum Scaled Value** and **(MX) Maximum Scaled Value** are used to specify the display range for the **(CV) Current Value**.

For example, if the **(CV) Current Value** is to be displayed as a percentage (0-100%) of a 10 VDC output range, set **(LS) Minimum Scaled Output** to 0 and **(HS) Maximum Scaled Output** to 100 (a display range of 0%-100% of full scale). Then set **(MN) Minimum Scaled Value** to 0.0 and **(MX) Maximum Scaled Value** to 100.0, so that when **(CV) Current Value** = 0 represents 0.0% of the output range and **(CV) Current Value** = 100 represents 100.0% of the output range.

#### 4.3.1.1 2-10VDC SCALING

If the output device in the previous example only operated from 2-10 V instead of 0-10 V, you would simply change the value of **(LS) Minimum Scaled Output** to be 20.0 because 2 V is 20% of the 10 V maximum. Everything else from the previous example would remain the same.

### 4.3.2 CONTROL MODE

**(CM) Control Mode** dictates how the Analog Output will be controlled. When set for 0 (Manual), the **(CV) Current Value** can be freely writable. When set for 1 (Automatic), the **(CV) Current Value** is controlled by a corresponding PID Control Loop object, available from the Controls Loop category of SoloPro.

When set for Automatic mode, attribute **(CB) Controlled By** provides accurate information relative to which process may currently be controlling the Analog Output. For example, if **(CM) Control Mode** = Auto, and configured Interlocking occurs, **(CB) Controlled By** will equal 3 (Interlocked Mode).

### 4.3.3 INPUT INTERLOCKING

Input Interlocking permits Analog Outputs to be set to a default position in the event that a Universal Input (configured as a Digital Input) has a positive value.

To configure Interlocking, select one or more inputs from **(IL) Inputs for Interlocking**. When multiple inputs are selected, the GPC operates in an *OR* fashion, where if any of the selected inputs contain a positive value, the Analog Output will be forced to a specific position.

The position that the Analog Output is commanded to is dictated through attribute **(IP) Interlock Position**.

### 4.3.4 ADDITIONAL INTERLOCK

The Additional Interlock feature permits the Analog Output to be interlocked through any defined Channel and Attribute in the controller, using **(IC) [Additional Interlock] Input Channel** and **(IA) [Additional Interlock] Input Attribute**. The triggering state of this additional interlock is controlled through attribute **(IT) [Additional Interlock] Trigger**.

### 4.3.5 ACTUAL OUTPUT VOLTAGE AND ACTUAL OUTPUT CURRENT

**(OV) Actual Output Voltage** and **(OC) Actual Output Current** defines the actual output value being sent by the GPC controller. This value can be cross-referenced to the present-value of the Analog Output

channel for I/O troubleshooting. Please note that while both voltage and current feedback are provided, the VI jumper determines which value is meaningful for any given Analog Output.

#### 4.3.6 COMMUNICATION FAILURE ENABLE POSITIONING

Communication Failure Positioning permits an Analog Output to be placed to a specific state should the GPC lose communications with the PUP network. This feature can be enabled through attribute **(CF) Communication Failure Enable = 1** (Yes).

Attribute **(FP) Comm Failure Position** dictates the value at which the Analog Output will default to in the event of a communication failure with the PUP network. This feature must be set up in the EE02 channel (Comm Failure)

#### 4.3.7 FIRE OVERRIDE

When a Universal Input has been configured for Fire Alarming, Analog Outputs can be configured to be set to a default position in the event of a fire alarm. Fire Override is controlled through attribute **(FE) Enable Fire Override**. This feature must be set up in the E001 channel (Fire Mode)

**(FI) Fire Position** dictates the value at which the Analog Output will default to in the event of an active sensed fire alarm by a properly configured input.

#### 4.3.8 UPDATE THRESHOLD

**(UT) Update Threshold** defines how often (in seconds) the GPC updates the actual output. By default, this value is set to 0.0, which commands the GPC to update the output immediately.

#### 4.3.9 RUN HOURS

**(RH) Run Hours** specifies how many hours the current value of the analog output has been non-zero.

## 4.4 DIGITAL OUTPUT SUMMARY

The Digital Output Summary channel provides a summary overview of all Digital Outputs on the GPC. Through this channel, a user can easily view key information regarding all of the Digital Output channel including the current value of all digital outputs as well as the actual output state. The Digital Output Summary channel has the following attributes: **ON**, **CV**, **AM**, **IP**, **CF**, **FE**, **FS**, **FI**, **IM**, **RE** and **OU**.

Attribute **ON** contains the name of the Summary Channel. This is a user definable string that helps to identify the channel.

Attribute **CV** displays the current state for each digital output (on or off). CV is a bitmap with bit #0 corresponding to DO1, bit #1=DO2, etc. up to bit #11=DO12. When the output is energized, the corresponding bit in CV will be set to 1.

Attribute **AM** determines which outputs will operate in automatic mode. **AM** is a bitmap with bit #0 corresponding to BO1, bit #1=BO2, etc. up to bit #11=BO12. If the corresponding bit in **AM** is set equal to 1, then the corresponding output is being controlled automatically by the SBC-GPC1. If a bit in **AM** is set equal to 0, then the corresponding output is being controlled manually.

Attribute **CF** is the communications failure enable/disable attribute. This attribute enables the communications failure feature on an individual basis for each digital output channel of the SBC-GPC. If the digital output is configured to operate in manual mode (**AM**=0), then the controller assumes the output states are being controlled by a host. If communications with the host are lost (bit #5 of FF00; **FA**=1), and **CF** is enabled by setting **CF**=1, then the current output value (**CV**) reverts to a programmed failure position (**FP**) for the corresponding output. **CF** is a bitmap with bit #0 corresponding to BO1, bit #1=BO2, etc. up to bit #11=BO12. Enabling a bit in **CF** will cause the corresponding output to assume the state defined in **FS** in the case of a communications failure.

Attribute **FS** defines the communication failure states for the digital outputs. **FS** functions in conjunction with **CF** and specifies the desired failure state for the corresponding input. **FS** is a bitmap with bit #0 corresponding to BO1, bit #1=BO2, etc. up to bit #11=BO12. The value of each bit in **FS** represents the desired failure state of the corresponding output (0=off, 1=on).

Attribute **FI** defines the fire states for the digital outputs. **FI** functions in conjunction with **FE** and specifies the desired failure state for the corresponding input. **FI** is a bitmap with bit #0 corresponding to BO1, bit #1=BO2, etc. up to bit #11=BO12. The value of each bit in **FI** represents the desired fire state of the corresponding output (0=off, 1=on).

Attribute **IM** specifies the Interlock position for the digital output. **IM** is a bitmap with bit #1 corresponding to DO1, etc. When the value in **IM** is equal to 1 the corresponding output will be set to an on state.

Attribute **FE** specifies which outputs will be enabled in the event of a fire. **FE** is a bitmap with bit #0 corresponding to BO1, bit #1=BO2, etc. up to bit #11=BO12. If the SBC-GPC1 enters fire mode (FF00; **FA**=1), you may wish to turn certain outputs either on or off. Setting a bit in **FE** to 1 will cause the corresponding output to assume the state defines in **FI** when a fire event is detected. If **FE**=0 for a given bit, then the output will not take any action when a fire is detected.

Attribute **RE** indicates channel reliability for the digital outputs. The reliability is only used for outputs on the STATbus. When the a control sequence within the SBC-GPC1 changes the value of an output, the SBC-GPC1 sends a command out over the STATbus network telling the output device to change its value.

If the message to change its value is successfully received, then the STATbus device will send a message back to the SBC-GPC1 acknowledging the request. If the SBC-GPC1 sends such a message, but does not receive an acknowledgement, then the controller will flag the output as unreliable. The channel reliability, **RE**, is a bitmap with bit #0 corresponding to BO1, bit #1=BO2, etc. up to bit #11=BO12. If an output is unreliable, the SBC-GPC1 will set the corresponding bit in **RE** to 1. A value of zero indicates that the output is reliable.

Attribute **OU** displays the actual state of each of the digital outputs. It is useful to have both because it is possible for the actual output state to differ from the current value because of various staging delays.

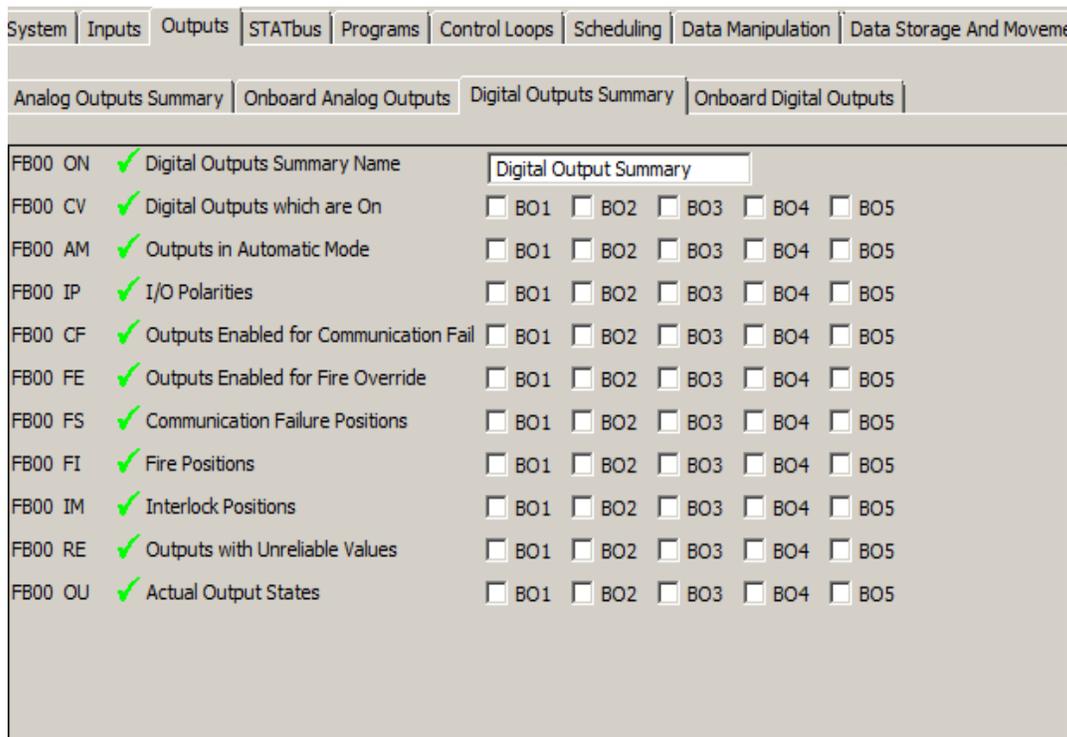


Figure 4-2 Digital Output Summary

## 4.5 DIGITAL OUTPUTS

Digital Outputs are used to provide on/off type control signals through use of on-board triacs or expansion outputs which can be either triac or relay based.

### 4.5.1 CONFIGURING MINIMUM OFF/ON TIMES

Digital Outputs can be programmed to maintain on/off signals through use of **(NT) Minimum On Time** and **(FT) Minimum Off Time**. These properties define how long in seconds, a binary output will maintain either remain off or on after it has been commanded to do so.

### 4.5.2 CONFIGURING POLARITY

The **(IP) Output Polarity** attribute defines the relationship between the physical state of the output and the software state reflected by **(CV) Current Value**. By default, **(IP) Output Polarity** is set to Normal (0). In this state, when the physical state of the output is inactive, the **(CV) Current Value** will indicate a value of inactive; and when the physical state of the output is active, the **(CV) Current Value** will indicate a value of active. If the **(IP) Output Polarity** were to be switched to Reverse (1), **(CV) Current Value** will indicate an Inactive status when the physical output is active, whereas **(CV) Current Value** would indicate an Active status when the physical output is inactive.

### 4.5.3 ACTUAL OUTPUT STATE

**(OU) Actual Output State** defines the physical output value being sent by the GPC controller. This value can be cross-referenced to the present-value of the Binary Output object for I/O troubleshooting.

### 4.5.4 CONTROL MODE

**(CM) Control Mode** dictates how the Digital Output will be controlled. When set for 0 (Manual), the **(CV) Current Value** can be freely writable. When set for 1 (Automatic), the **(CV) Current Value** is controlled by a corresponding TSTAT Loop object, available from the Controls Loop category of SoloPro. In Automatic mode, the output is also controlled by scheduling or via STAT overrides.

When set for Automatic mode, attribute **(CB) Controlled By** provides accurate feedback information relative to which process may currently be controlling the Digital Output. For example, if **(CM) Control Mode = Auto**, and configured Interlocking occurs, **(CB) Controlled By** will equal 3 (Interlocked Mode)

### 4.5.5 INPUT INTERLOCKING

Input Interlocking permits Digital Outputs to be set to a default position in the event that an Universal Input (configured as a Digital Input) has a positive value.

To configure Interlocking, select one or more inputs from **(IL) Inputs for Interlocking**. When multiple inputs are selected, the GPC operates in an *OR* fashion, where if any of the selected inputs contain a positive value, the Digital Output will be forced to an active state.

### 4.5.6 ADDITIONAL INTERLOCK

The Additional Interlock feature permits the Digital Output to be interlocked through any defined Channel and Attribute in the controller, using **(IC) [Additional Interlock] Input Channel** and **(IA) [Additional Interlock] Input Attribute**. The behavior of this additional interlock is controlled through attribute **(IT) [Additional Interlock] Trigger**.

#### 4.5.7 SCHEDULES TO FOLLOW

Digital Outputs may be directly controlled by a configured Schedule, as well as a Schedule override from a connected SBC-STAT product. **(SM) Schedule Map** provides a bitmap which can be used to select one or more Schedule channels, an Override from a connected STAT, the Host Schedule setting (F900;HO), or Through the Occupancy Extension channel (FC01).

When multiple items are selected, this feature works in an *OR* fashion, where if any of the selected schedules are occupied, the Digital Output will be commanded to an active state.

#### 4.5.8 INRUSH CURRENT LOCKOUT

The Inrush Current Lockout provides the ability to lockout specific Digital Outputs from one another through the use of **(CT) Inrush Current Lockout Delay Timer**, and **(CL) Inrush Current Lockout Map**.

**(CL) Inrush Current Lockout Map** allows you to select which digital outputs are considered to be in the same zone and will be affected by the inrush lockout. When digital outputs are selected, after the first one is commanded on, the additional outputs will be delayed on as specified in the **(CT) Inrush Current Lockout Delay Timer**.

**(CT) Inrush Current Lockout Delay Timer** is the time in seconds imposed before the digital outputs specified in the lockout map will be staged on.

#### 4.5.9 PULSE WIDTH

**(PW) Pulse Width when Output Is On** defines how often (in seconds) the Digital Output will pulse between on and on states to perform pulse width control when the output is driven to be active.

#### 4.5.10 RUN HOURS

**(RH) Run Hours** specifies how many hours the Digital Output has been a logical 1.

---

# SECTION 5: EXPANSION I/O

*This section describes the process of setup and configuration for working with expansion I/O products utilizing STATbus technology. Each expansion product is reviewed, along with the process for mapping them to GPC1 and GPC3 products.*

## IN THIS SECTION

What are IOX Modules?.....	5-3
Features of IOX Modules .....	5-3
Remote I/O and Mapping Points .....	5-3
IOX Module Specifications .....	5-4
General .....	5-4
SSB-FI1 .....	5-4
SSB-UI1 .....	5-4
SSB-AO1 .....	5-4
SSB-DI1 .....	5-5
SSB-DO1-I .....	5-5
SSB-DO1-I .....	5-5
SSB-DO2 .....	5-5
SSB-DO2-I .....	5-5
SSB-IOX1-1 .....	5-6
SSB-IOX1-2 .....	5-6
SSB-IOX2-1 .....	5-6
SSB-IOX2-2 .....	5-6
Length of the Network .....	5-8
Number of Devices .....	5-9
Communications Limits .....	5-9
GID Numbers and Mapping IOX Modules.....	5-11
Writing GIDs to Devices .....	5-11
Removing GID assignments .....	5-11
SSB-FI1 .....	5-13
SSB-UI1 .....	5-18
SSB-AO1 .....	5-25
SSB-DI1 .....	5-31
SSB-DO1 .....	5-36
SSB-DO1-I .....	5-40
SSB-DO2 .....	5-45
SSB-DO2-I .....	5-49
SSB-IOX1-x .....	5-55
SSB-IOX2-x .....	5-64



## 5.1 WHAT ARE IOX MODULES?

IOX modules are specialized STATbus devices which allow you to add remote I/O points to controllers in the GPC family.

IOX modules can be used with GPC1 and GPC3 products. GPC2 products may take on STAT/RHT devices only. These units have on-board I/O and, include the ability to add additional I/O, using IOX modules to achieve the number of inputs and outputs needed. This allows you to craft a controller with a completely customized I/O profile. In this way, you can tailor the controller to suit the job rather than designing the job around the capabilities of a controller. This will allow you to make decisions based on good design principles rather than system limitations.

### 5.1.1 FEATURES OF IOX MODULES

- . Provide remote I/O points to GPC controllers
- . Provide the ability to locate I/O hardware where it is most convenient
- . Communication via STATbus
- . Easy 2- or 4- wire connection using twisted pair wire
- . Easy configuration within GPC using SoloPro.

### 5.1.2 REMOTE I/O AND MAPPING POINTS

IOX modules provide additional, remote I/O points to GPC controllers. Modules exist that can provide additional Universal Input, Pulse Input, Analog Output and Digital Output points. These points appear to the GPC to be identical to an on-board input or output, therefore only minimal additional work is needed when commissioning IOX modules

Remote I/O behaves in the same way as on-board I/O, except that it is located remotely from the controller. Because they are not on-board, each remote device requires a unique address, known as a Global Identification (GID) number so that the controller may recognize and direct communications to it. When working with IOX modules, there is the additional commissioning step of associating the remote I/O point with inputs and outputs within the controller. This is accomplished by assigning the GID number of the device to the desired input or output along with specifying the Input Index (I#) or Output Index (O#). These indicate which physical point on the expansion device is being mapped to the I/O point on the GPC. Once the IOX module is mapped in this way, it will function as any other input or output of the same type.

#### NOTE

GPC Family devices only utilize Digital SBC-STAT's which are referenced in the following sections. Analog SBC-STAT devices are not compatible with the GPC controllers. For more information on the SBC-STAT family devices, please reference the *STAT User Manual*.

## 5.2 IOX MODULE SPECIFICATIONS

### 5.2.1 GENERAL

#### 5.2.1.1 NETWORKING

- . **communications protocol:** STATbus
- . **wiring:** 2- or 4-wire (device dependent), 18-22 ga., twisted pair
- . **update frequency:** nominally every 100 mS
- . **network configuration:** multidrop bus

#### 5.2.1.2 TERMINATIONS

- . Pluggable terminal blocks for inputs and/or outputs, power and network connection.

#### 5.2.1.3 OPERATING ENVIRONMENT

- . **temperature range:** 32-122°F (0-50°C)
- . **humidity range:** 0-80% RH, non-condensing

#### 5.2.1.4 AGENCY APPROVALS

- . UL listed 916, Management Equipment, Energy (PAZX)
- . FCC rules Part 15 Class B Computing Device
- . Complies with CE directives and standards (XAPX2)

### 5.2.2 SSB-FI1

#### 5.2.2.1 I/O

- . One (1) 12-bit Universal Input (interpolated to a 16-bit value)
- . Selectable 0-5 VDC, 0-10 VDC, 0-20 mA or 0-250 kΩ input range

#### 5.2.2.2 POWER REQUIREMENTS

- . None

#### 5.2.2.3 DIMENSIONS

- . **size:** 3.02 x 1.41 x 0.95in. (7.67 x 3.58 x 2.41 cm)
- . **shipping weight:** .04 lb. (.018 kg)

### 5.2.3 SSB-UI1

#### 5.2.3.1 I/O

- . One (1) 24-bit Universal Input
- . Selectable 0-5 VDC, 0-10 VDC, 0-20 mA or 0-250 kΩ input range

#### 5.2.3.2 POWER REQUIREMENTS

- . 24VAC, 50/60 Hz, 1 A (max)

#### 5.2.3.3 DIMENSIONS

- . **size:** 4.2 x 4.2 x 1.0 in. (10.67 x 10.67 x 2.54 cm)
- . **shipping weight:** .50 lb. (.23 kg)

### 5.2.4 SSB-AO1

#### 5.2.4.1 I/O

- . One (1) Analog Output
- . Selectable 0-10 VDC or 0-20 mA output range

#### 5.2.4.2 POWER REQUIREMENTS

- . 24VAC, 50/60 Hz, 1 A (max)

#### 5.2.4.3 DIMENSIONS

- . **size:** 4.2 x 4.2 x 1.0 in. (10.67 x 10.67 x 2.54 cm)
- . **shipping weight:** .50 lb. (.23 kg)

### 5.2.5 SSB-DI1

#### 5.2.5.1 I/O

- One (1) Optically Isolated, Pulse Counting, Digital Input

#### 5.2.5.2 POWER REQUIREMENTS

- 24VAC, 50/60 Hz, 1 A (max)

#### 5.2.5.3 DIMENSIONS

- **size:** 4.2 x 4.2 x 1.0 in. (10.67 x 10.67 x 2.54 cm)
- **shipping weight:** .50 lb. (.23 kg)

### 5.2.6 SSB-DO1

#### 5.2.6.1 I/O

- One (1) Digital Output (relay)

#### 5.2.6.2 POWER REQUIREMENTS

- 24VAC, 50/60 Hz, .25 A (max)

#### 5.2.6.3 DIMENSIONS

- **size:** 4.75 x 3.25 x 2.0 in. (12.07 x 8.26 x 5.08 cm)
- **shipping weight:** .50 lb. (.23 kg)

### 5.2.7 SSB-DO1-I

#### 5.2.7.1 I/O

- One (1) Digital Output (relay)
- One (1) Digital Input (dry contact only, no pulse counting)

#### 5.2.7.2 POWER REQUIREMENTS

- 24VAC, 50/60 Hz, .25 A (max)

#### 5.2.7.3 DIMENSIONS

- **size:** 4.75 x 3.25 x 2.0 in. (12.07 x 8.26 x 5.08 cm)
- **shipping weight:** .50 lb. (.23 kg)

### 5.2.8 SSB-DO2

#### 5.2.8.1 I/O

- Two (2) Digital Outputs (relays)

#### 5.2.8.2 POWER REQUIREMENTS

- 24VAC, 50/60 Hz, .25 A (max)

#### 5.2.8.3 DIMENSIONS

- **size:** 4.75 x 3.25 x 2.0 in. (12.07 x 8.26 x 5.08 cm)
- **shipping weight:** .56 lb. (.25 kg)

### 5.2.9 SSB-DO2-I

#### 5.2.9.1 I/O

- Two (2) Digital Outputs (relays)
- Two (2) Digital Inputs (dry contacts only, no pulse counting)

#### 5.2.9.2 POWER REQUIREMENTS

- 24VAC, 50/60 Hz, .25 A (max)

#### 5.2.9.3 DIMENSIONS

- **size:** 4.75 x 3.25 x 2.0 in. (12.07 x 8.26 x 5.08 cm)
- **shipping weight:** .56 lb. (.25 kg)

**5.2.10 SSB-IOX1-1****5.2.10.1 I/O**

- . Four (4) 24-bit Universal Inputs
- . Selectable 0-5 VDC, 0-10 VDC, 0-20 mA or 0-250 k $\Omega$  input range
- . One (1) Optically Isolated, Pulse Counting, Digital Input
- . Two (2) Analog Outputs
- . Selectable 0-10 VDC or 0-20 mA output range
- . Two (2) Digital Outputs (triacs)

**5.2.10.2 POWER REQUIREMENTS**

- . 24VAC, 50/60 Hz, 1.85 A (max)

**5.2.10.3 DIMENSIONS**

- . **size:** 5.75 x 6.35 x 1.05 in. (14.60 x 16.13 x 2.67 cm)
- . **shipping weight:** .95 lb. (.42 kg)

**5.2.11 SSB-IOX1-2****5.2.11.1 I/O**

- . Eight (8) 24-bit Universal Inputs
- . Selectable 0-5 VDC, 0-20mA or -250 k $\Omega$  input range.

**5.2.11.2 POWER REQUIREMENTS**

- . 24VAC, 50/60 Hz, 1.85 A (max)

**5.2.11.3 DIMENSIONS**

- . **size:** 5.75 x 6.35 x 1.05 in. (14.60 x 16.13 x 2.67 cm)
- . **shipping weight:** .95 lb. (.42 kg)

**5.2.12 SSB-IOX2-1****5.2.12.1 I/O**

- . Twelve (12) 24-bit Universal Inputs
- . Selectable 0-5 VDC, 0-10 VDC, 0-20 mA or 0-250 k $\Omega$  input range
- . Six (6) Analog Outputs
- . Selectable 0-10 VDC or 0-20 mA output range
- . Six (6) Digital Outputs (triacs)

**5.2.12.2 POWER REQUIREMENTS**

- . 24VAC, 50/60 Hz, 1.85 A (max)

**5.2.12.3 DIMENSIONS**

- . **size:** 8.407 x 6.5 x 1.25 in. (20.83x16.51x3.18 cm)
- . **shipping weight:** 3 lb. (1.36 kg)

**5.2.13 SSB-IOX2-2****5.2.13.1 I/O**

- . Twelve (12) 24-bit Universal Inputs
- . Selectable 0-5 VDC, 0-10 VDC, 0-20 mA or 0-250 k $\Omega$  input range

**5.2.13.2 POWER REQUIREMENTS**

- . 24VAC, 50/60 Hz, 1.85 A (max)

**5.2.13.3 DIMENSIONS**

- . **size:** 8.407 x 6.5 x 1.25 in. (20.83x16.51x3.18 cm)
- . **shipping weight:** 3 lb. (1.36 kg)

**5.2.14 SSB-STAT1D, SBC-STAT2D, SBC-STAT3****5.2.14.1 I/O**

- . One (1) Digital Temperature Sensor
- . 50° to 122°  $\pm$  0.9°F (10° to 50°  $\pm$  0.5°C)

## 5.2.14.2 POWER REQUIREMENTS

- . None

## 5.2.14.3 DIMENSIONS

- . **size:** 2.78 x 3.86 x 1.03 in. (7.06 x 9.80 x 2.62 cm)
- . **weight:** 4.0oz (113.5g)
- .

## 5.2.15 SBC-RH1, SBC-RH3

## 5.2.15.1 I/O

- . One (1) Digital Humidity Sensor
- . 0 to 100%  $\pm$  2%

## 5.2.15.2 POWER REQUIREMENTS

- . None

## 5.2.15.3 DIMENSIONS

- . **size:** 2.78 x 3.86 x 1.03 in (7.06 x 9.80 x 2.62 cm)
- . **weight:** 4.0oz (113.5g)

## 5.2.16 SBC-RHT

## 5.2.16.1 I/O

- . One (1) Digital Temperature Sensor
- . 50° to 122°  $\pm$  0.9° (10° to 50°  $\pm$  0.5°C)
- . One (1) Digital Humidity Sensor
- . 0 to 100%  $\pm$  2%

## 5.2.16.2 POWER REQUIREMENTS

- . None

## 5.2.16.3 DIMENSIONS

- . **size:** 2.78 x 3.86 x 1.03 in (7.06 x 9.80 x 2.62 cm)
- . **weight:** 4.0oz (113.5g)

### 5.3 LENGTH OF THE NETWORK

The distance measured from the controller to the STATbus device on the network located furthest away from it should not exceed 1000' in length. The STATbus shown in Figure 5-1a is a valid configuration because the distance from the controller to the most distant device is less than 1000' whereas the configuration shown in Figure 5-1b is not valid because the total length to the most distant device is 1150', exceeding the 1000' maximum.

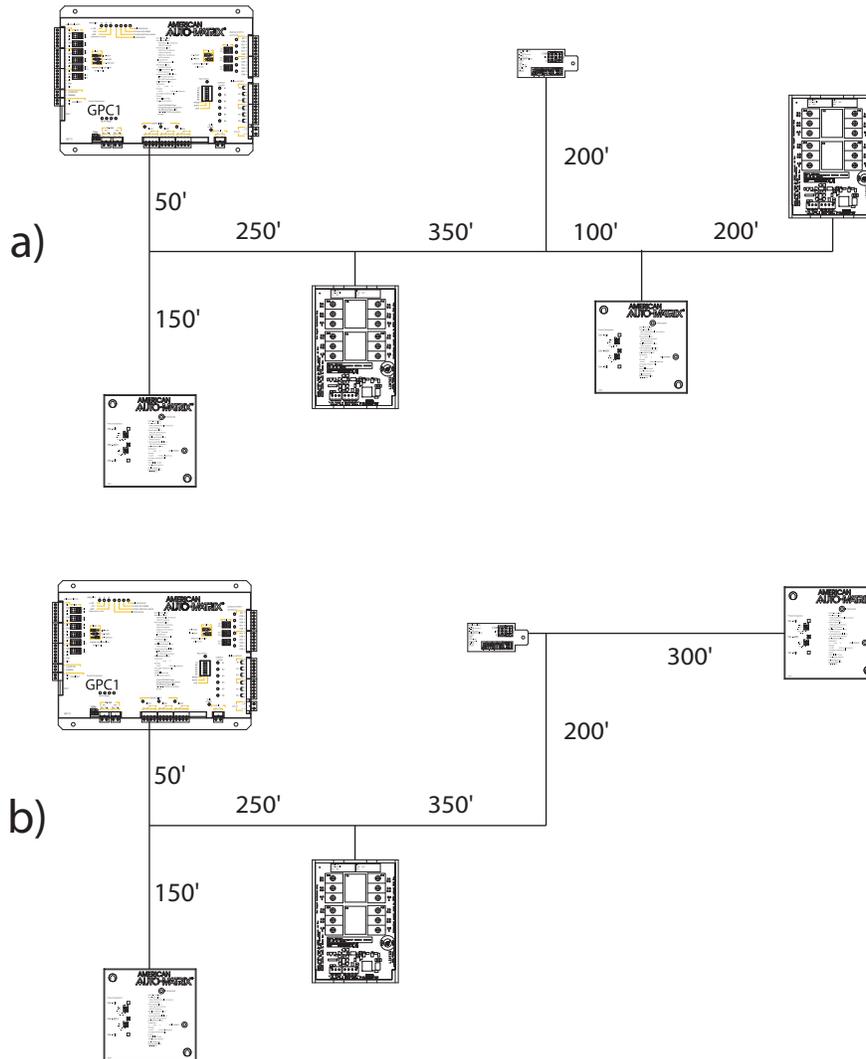


Figure 5-1 Determining Maximum STATbus Length

## 5.4 NUMBER OF DEVICES

Each STATbus channel on the controller will support a maximum of thirteen (13) physical devices.

**CAUTION**



*A maximum of twelve (12) devices can be connected to a single STATbus channel. If more than twelve devices are connected, only twelve will be enumerated by the controller. If more than twelve devices are connected, there is no way to predict which devices will be enumerated and which will be left off.*

### 5.4.1 COMMUNICATIONS LIMITS

While the STATbus protocol allows up to thirteen devices to be connected to a single network, certain devices reduce the maximum number of other devices that may be used on a single channel. In particular, SBC-STAT's have a higher power requirement than other STATbus devices and limit the total number of devices that can be put on the network and still communicate. When one or more STAT devices are included on the network, the maximum number of devices allowed on the network will be reduced.

#### 5.4.1.1 NO STATs ON THE STATBUS

If your STATbus channel does not have any STATs (STAT1D, STAT2D, or STAT3) on it, then you may have up to thirteen devices in any combination on the network. This may include SSB-FI1s, SSB-UI1s, SSB-AO1s, SSB-DI1s, SSB-DO1s, SSB-DO1-Is, SSB-DO2s, SSB-DO2-Is, and SSB-IOX1 modules.

#### 5.4.1.2 ONE OR MORE STATs ON THE STATBUS

If one or more STATs are being used, the total number of devices that can communicate on a single STATbus channel will be reduced. Table 5-1 lists the number of additional STATbus devices that can be connected for a given number of STATs.

*Table 5-1 Number of Devices Allowed on a STATbus*

Number of SBC-STAT's	Number of other STATbus Devices
0	12
1	10
2	8
3	6
4	4
5	2

#### 5.4.1.3 EXAMPLE: NO STATs ON THE STATBUS

With no STATs on the Bus, you may use any combination of SSB devices, up to a maximum of twelve devices total. This means any of the following would be valid:

- . 12 SSB-FI1s to read a number of different inputs

- . 6 SSB-FI1s and 6 SSB-AO1s to provide simple damper control for six zones
- . 3 SSB-UI1s, 3 SSB-AO1s, and 3 SSB-DO1-Is, giving three zones with a zone temperature input, control for a damper and a reheat coil with supervisory monitoring

#### 5.4.1.4 EXAMPLE: STATS ON THE STATBUS

For a STATbus design that contains STATS, you must refer to the Table 5-1 above to determine the number of devices that can be used in addition to the STATS.

For a system with 4 STATS, for example, Table 5-1 indicates that up to four additional devices can be connected to the bus. You could use four SSB-AO1s to create four zones with damper control.

## 5.5 GID NUMBERS AND MAPPING IOX MODULES

### 5.5.1 WRITING GIDS TO DEVICES

Every IOX module has a unique Global Identification (GID) which is identical to the unit's serial number. The GID number is the address that the GPC will use to uniquely identify and communicate with the module. The GID numbers are used during commissioning to map the remote I/O point(s) on the IOX modules to inputs and/or outputs in the GPC.

#### NOTE



It is recommended that, at some point before or during installation, you compile a list of all the STATbus devices, their location or function, and their GID numbers. This information will be needed in the commissioning of the project and may be difficult to obtain once the units are installed.

To prepare the controller to perform this mapping you must set the **(CR) Configure Remote I/O** attribute under STATbus>UI Mapping area to “Edit I/O GIDs” (**CR=2**). This allows you to manually assign the GID numbers of remote I/O devices located on the STATbus to the inputs or outputs in the GPC.

You must select an input or output in the controller and choose the device you wish to assign to it. You must then enter the GID number of the chosen device in to the **(GI) GID of I/O Device** attribute for the input or output. If the GID entered is valid, the GID number will be displayed, along with the type of device.

Once **(GI) GID of I/O Device** has been set, you must then configure the index number properties **(I#) Input Index of I/O Device** for input objects, and **(O#) Output Index for I/O Device** for output objects. For IOX modules that only contain a single I/O point, the index number shall be set to a value of 1. For IOX modules that contain multiple points of data, you will need to set the index number according to the I/O assignment. For example, if you wish to configure a Universal Input to focus on UI2 of an SSB-IOX, set the index number to a value of 2. For an RHT STAT Input Index 1 is temperature and Input Index 2 is humidity. Each IOX module reviewed will include a table that provides a reference for the index number that corresponds with each input or output.

Repeat this process of assigning GIDs for as many devices as you wish to configure.

Once all the devices you wish to configure have had their GIDs successfully assigned to an input or output, you must set the **(CR) Configure Remote I/O** = 1. This will write the configuration information to the devices on the STATbus network. The Configure Remote I/O setting will automatically return to “Normal” (**CR=0**) once the write is complete, eliminating the possibility of accidentally overwriting STATbus configuration information.

### 5.5.2 REMOVING GID ASSIGNMENTS

Once the GID of an IOX module has been mapped to a particular input or output, that mapping is stored both in the controller and on the device itself. If you wish to remove a GID mapping, you can do so by entering a value of 0 into the **GI** attribute or attribute for the input or output. Unmapping a module in this way will only remove the assignment to that particular input or output and will not effect any other inputs or outputs to which the module is mapped. If the module has multiple inputs or outputs, this will leave all other mappings intact. This situation will cause communication problems between the controller and the module

that will result in the module behaving unpredictably. Additionally, **I#** and **O#** can be set to a value of 255 to unassign the device.

**NOTE**

When unmapping the GID of a module with multiple inputs and/or outputs, you must zero the GID in all objects or channels to which it is assigned.

## 5.6 SSB-FI1

### 5.6.1 FEATURES

The SSB-FI1, shown in Figure 5-2, is a STATbus device which provides a single remote Flexible Input, that is installed at the sensor location - in all most cases, directly coupled with the sensor. A Flexible Input is a lower resolution version of the Universal Input found on the GPC itself. The signal read by the SSB-FI1 is processed using a 12-bit analog-to-digital converter and, through digital signal processing algorithms, extrapolated to a 16-bit reading.

The SSB-FI1 is an option for sensors which do not require excitation power or for inputs which do not require the additional resolution provided by the SSB-UI1.

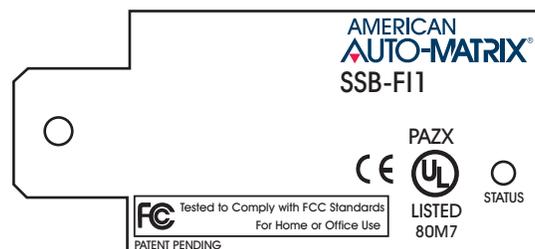
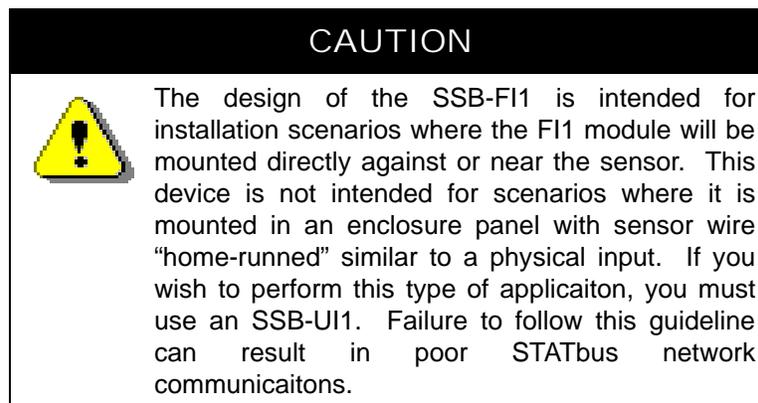


Figure 5-2: The SSB-FI1

The input on the SSB-FI1 can be configured to read a 0-10 V, 0-5 V, 0-20 mA or 0-250 k $\Omega$  signal using jumpers located on the device. Any of the jumper configurations can also be interpreted by the controller as a digital value by setting the sensor type to digital (**ST=0**). When being interpreted as a digital signal, a reading of 0-25% of full scale corresponds to a zero state and a reading of 26-100% of full scale corresponds to a one state.

### 5.6.2 WIRING/CONFIGURATION

#### 5.6.2.1 IVR JUMPER

Before installing the SSB-FI1, you must configure the SSB-FI1 for the type of sensor connected to it, connect the sensor to be used, and connect the SSB-FI1 to the STATbus network. The jumpers used to determine the type of sensor connected to the SSB-FI1 are shown in Figure 5-3.

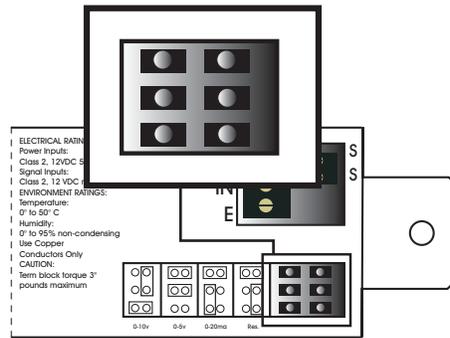


Figure 5-3: Location of the Input Select Jumpers

Once the type of sensor that will be connected to the SSB-FI1 has been determined, the jumpers should be moved to the positions appropriate for that type. The jumper settings for a 0-10 V, 0-5 V, 0-20 mA and 0-250 kΩ resistive inputs are given in Figure 5-4a-d respectively. The jumper configurations are also printed on the SSB-FI1 enclosure adjacent to the jumpers.

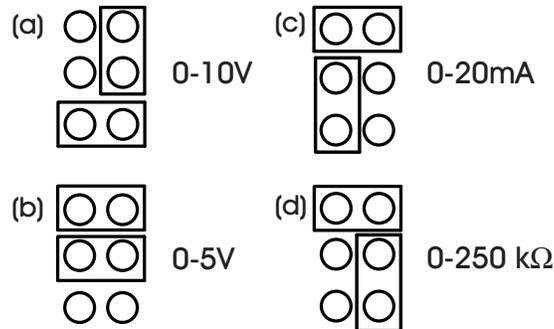


Figure 5-4: Jumper Settings for the SSB-FI1

### 5.6.2.2 INPUT WIRING

The SSB-FI1 is ideal for any sensor which does not require power. A voltage sensor (0-10 V or 0-5V), current sensor (0-20 mA or 4-20 mA), or resistance sensor would all be wired to the SSB-FI1 by connecting the common wire to the COM terminal and the signal wire to the IN terminal as shown in Figure 5-5.

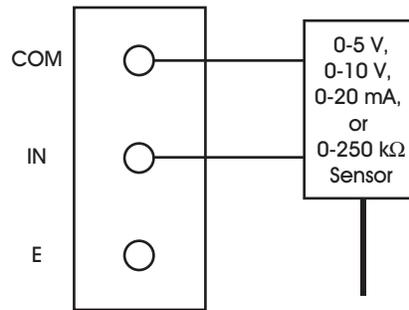


Figure 5-5: Wiring an Input to the SSB-FI1

### NOTE



The COM terminal on the SSB-FI1 is not an electrical ground. The SSB-FI1 will only function properly when connected to isolated, non-grounded sensors.

#### 5.6.2.3 POSITION POTENTIOMETER

The SSB-FI1 can also be used to read information from a position potentiometer. For this type of sensor, the jumpers must be set so that the SSB-FI1 is configured as a voltage input (either 0-5 V or 0-10 V). One side of the resistor should be connected to the COM terminal and the other side to the E terminal. The wiper, providing the position information, is connected to the IN terminal as shown in Figure 5-6.

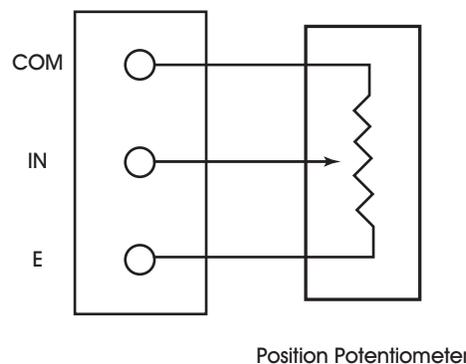


Figure 5-6: Wiring a Position Potentiometer to the SSB-FI1

Once the sensor is connected to the SSB-FI1, it must be added to the STATbus network. Connect the pair of wires coming from the last STATbus device to the two-pin terminal block labelled SS on the SSB-FI1. The STATbus network is non-polar, so you do not need to worry about maintaining polarity between devices.

### 5.6.3 MOUNTING THE SSB-FI1

The SSB-FI1 is designed to be mounted inside a standard 2x4 junction box as shown in Figure 5-7. The SSB-FI1 is mounted in the junction box by attaching a screw to the junction box, through the mounting hole, securing the SSB-FI1. The SSB-FI1 should be mounted such that the terminal blocks face the inside of the junction box. When the SSB-FI1 is correctly installed, the GID number printed on the label should be clearly visible.

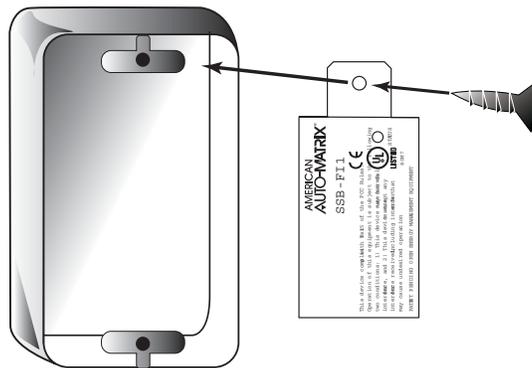
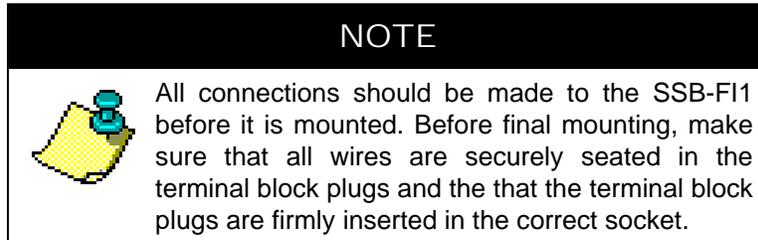


Figure 5-7: Mounting the SSB-FI1

### 5.6.4 STATUS INDICATOR LED

The SSB-FI1 has a status indicator LED which provides feedback as to the device's current operational status. The status indicator LED is located on the front of the SSB-FI1 (the side that faces out when installed) as shown in Figure 5-8. This allows status diagnostics to be performed without having to remove the SSB-FI1 or disconnect it from the STATbus.

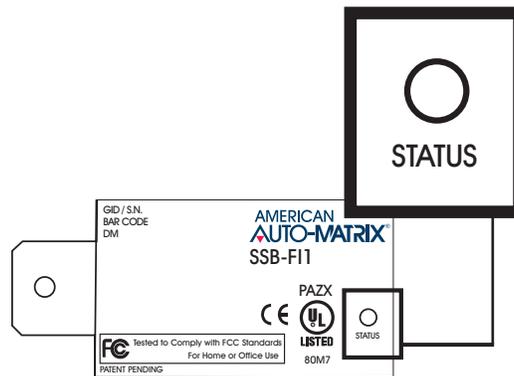


Figure 5-8: Location of the Status Indicator LED on the SSB-FI1

The status indicator LED shows one of four different states: powered but not enumerated, enumerated but not configured, configured, and “identify”. The different states are indicated by the rate at which the LED blinks. The LED blinking quickly, approximately three to four times per second, indicates that the unit is powered but has not yet been enumerated by the controller. This is useful for identifying units that are correctly wired but not configured. When the device is enumerated but not configured, the blink rate will slow to approximately twice a second.

Once the device has been configured, the blink rate will slow down to approximately one blink per second. This will be the normal state of the device when it is correctly wired, powered, enumerated, and configured in the controller.

When the controller is set to “Identify” in the Configure Function and Configure Device properties, the LED on the SSB-FI1 will be blink three times in quick succession and then pause before repeating the three blinks again. This is especially useful for quickly identifying an individual device in the field when troubleshooting the STATbus.

### 5.6.5 SSB-FI CONFIGURATION TABLE

Table 5-2: SSB-FI Configuration Table

I/O Quantity	(I#) Index Number
1 Universal Input	1

## 5.7 SSB-UI1

### 5.7.1 FEATURES

The SSB-UI1, shown in Figure 5-9, is a STATbus device which provides a single remote universal input. The input the SSB-UI1 provides is a true 24-bit universal input with more robust signal processing electronics than the SSB-F11. The SSB-UI1 should be used when using an input which requires excitation power. The SSB-F11, while capable of reading the signal from such devices, albeit at a lower resolution, is not capable of providing excitation power, so it is not a viable choice when dealing with sensors of this kind.



<p>PRODUCT DESIGNATION:</p> <p><b>SSB-UI1</b> </p> <p>I: 0-20mA </p> <p>V: 0-10VDC </p> <p>R: 0-250K<math>\Omega</math> </p> <p>JUMPER CONFIGURATION (TYPICAL UI)</p> <p><b>SSB-AO1</b> </p> <p>V: 0-10VDC </p> <p>I: 0-20mA </p> <p>JUMPER CONFIGURATION (TYPICAL FOR AO)</p> <p><b>SSB-DI1</b> </p> <p><small>This device complies with Part 15 of the FCC Rules. Operation of this equipment is subject to the following two conditions: 1) This device may not cause harmful interference, and 2) This device must accept any interference received, including interference that may cause undesired operation. (0579b) PATENT PENDING</small></p>	<p>IO PROCESSOR</p> <p>ELECTRICAL RATINGS:</p> <p>Power Inputs: Class 2 24VAC 50/60 Hz Nominal, .65A maximum .2A PTC Protection</p> <p>Power Outputs: Class 2 24 VDC OUT, .065A maximum .065A PTC Protection</p> <p>Signal Inputs/Outputs: Class 2 30 VDC maximum</p> <p>Communication Signals: Class 2 SSB, 12VDC, 50mA maximum</p> <p>ENVIRONMENT RATINGS:</p> <p>Temperature: 0° to 50° C</p> <p>Humidity: 0 to 80% RH non-condensing</p> <p>Use Copper Conductors Only</p> <p>Caution: Terminal Block Torque 3 Inch-pounds maximum</p> <p>For Instructions See Reference Manual No. IE-04-00-0126</p> <p>Caution: Improper wiring will cause damage. Refer to Manual</p> <p style="text-align: center;"><b>MAIN POWER</b></p> <p style="text-align: center;">80M7, OPEN ENERGY MANAGEMENT EQUIPMENT</p>
--	---




Figure 5-9: The SSB-UI1

### 5.7.2 WIRING/CONFIGURATION

Connecting a sensor to the SSB-UI1 requires two steps, connecting the wires from the sensor to the SSB-UI1 and configuring the IVR jumper.

Connections are made to the SSB-UI1 via the terminal blocks, located on the side of the unit which faces into the junction box, as shown in Figure 5-10. There are two sets of terminal blocks, the first is for connection to the STATbus network and power and the second is for the connection of the sensor to the SSB-UI1.

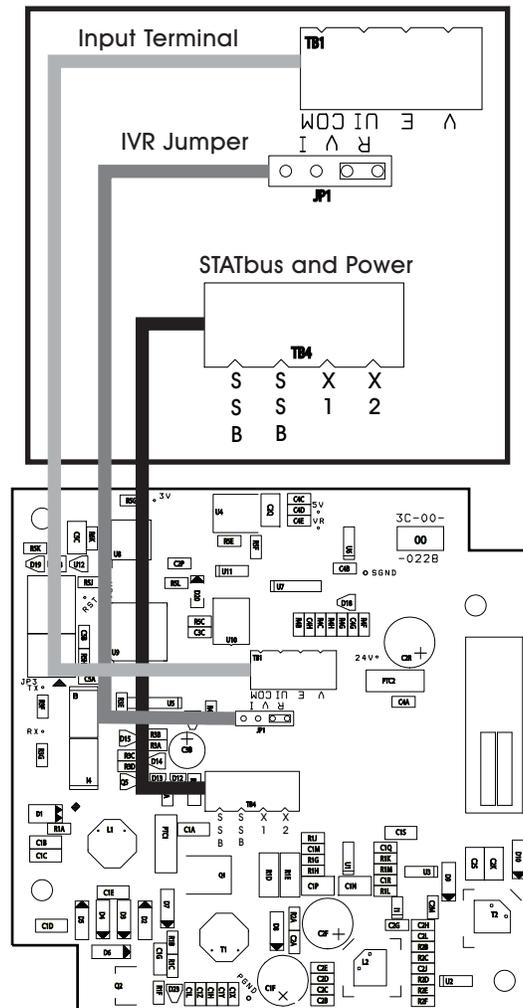


Figure 5-10: SSB-UI1 Terminal Block Locations

#### 5.7.2.1 IVR JUMPER

The SSB-UI1 has an IVR jumper, identical in function to the ones found on the GPC, which is used to select the type of input connected to the module. The SSB-UI1 can be configured to read a 0-20 mA, 0-10 V or a 0-250 k $\Omega$  sensor. The jumper settings corresponding to these options are shown in Figure 5-11a-c respectively.

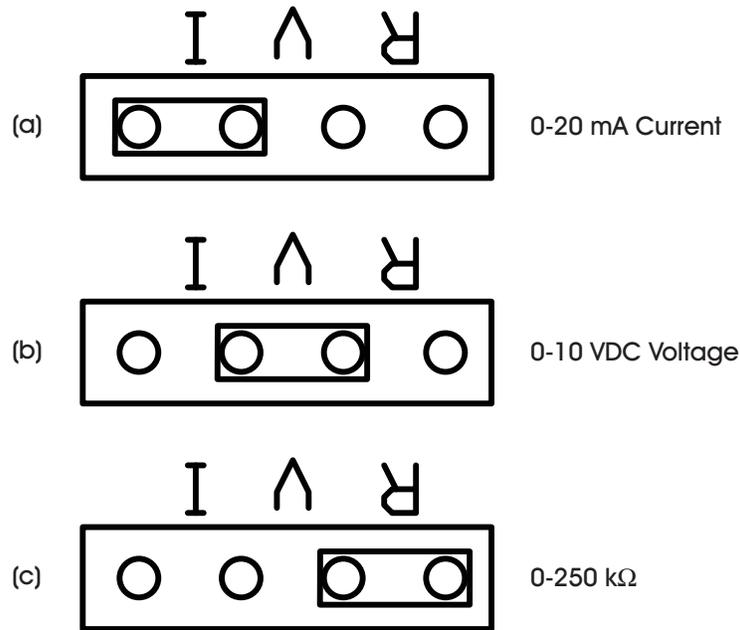


Figure 5-11: SSB-UI1 IVR Jumper Positions

#### 5.7.2.2 RESISTIVE INPUTS

A resistive input, such as a thermistor, would be connected as shown in Figure 5-12. One side of the input should be connected to the UI terminal and the other to the COM terminal. Since a thermistor is a resistive input, the IVR jumper should be set to the “R” position.

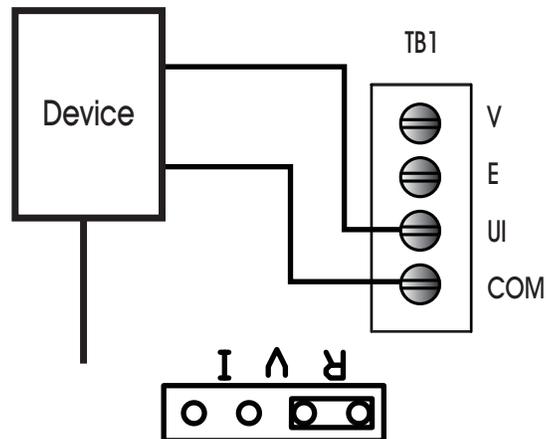


Figure 5-12: Wiring a Resistive Input to the SSB-UI1

#### 5.7.2.3 VOLTAGE INPUTS

The connections needed to use a voltage sensor with the SSB-UI1 are shown in Figure 5-13. The signal wire from the sensor should be connected to the UI terminal, the power connection should be connected to

the V terminal and the common wire should be connected to the COM, terminal V terminal provides 24VDC output. The IVR jumper should be set to the "V" position when using this type of input.

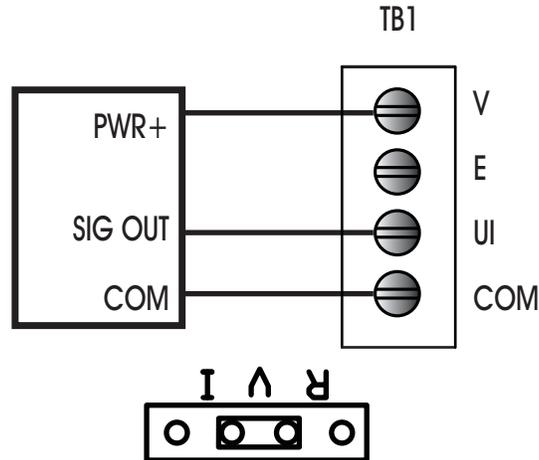


Figure 5-13: Wiring a Voltage Input to the SSB-UI1

#### 5.7.2.4 CURRENT INPUTS

When using a current sensor with the SSB-UI1, the sensor would be connected as shown in Figure 5-14. The + and - terminals of the sensor should be connected to the V and UI terminals on the SSB-UI1 respectively. When using 3-wire current sensors, the COM terminal on the sensor should be connected to the COM terminal on the SSB-UI1. Regardless of which type of current sensor you are using, the IVR jumper should be set to the "I" position.

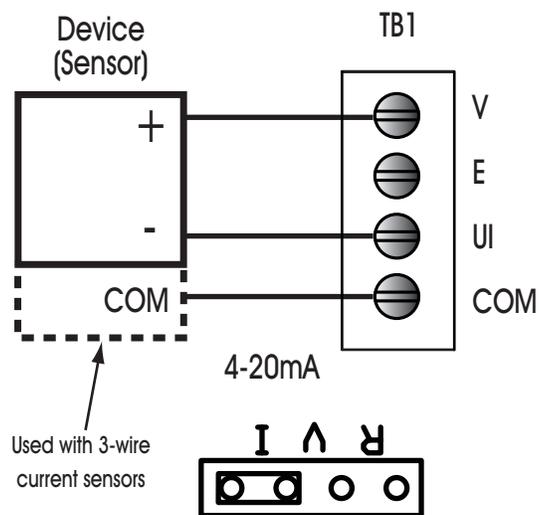


Figure 5-14: Wiring a Current Input to the SSB-UI1

### 5.7.2.5 POSITION POTENTIOMETER

The SSB-UI1 can be configured to read the signal from a position potentiometer as shown in Figure 5-15. For this type of sensor, the one side of the resistor should be connected to the COM terminal, the other side should be connected to the E terminal, and the wiper, providing the position information, should be connected to the UI terminal. When using this type of sensor, the IVR jumper should be set to the "V" position.

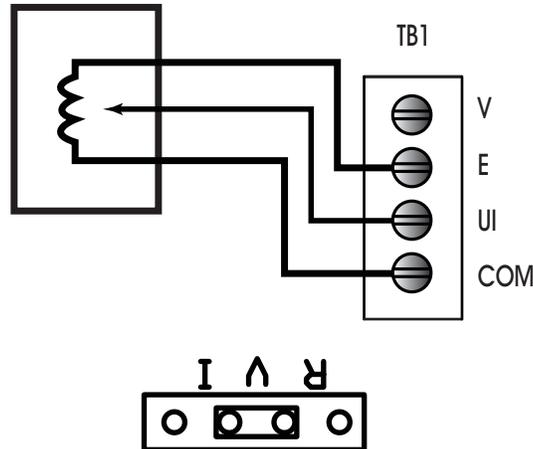


Figure 5-15: Wiring a Position Potentiometer to the SSB-UI1

### 5.7.3 MOUNTING THE SSB-UI1

The SSB-UI1 has the same footprint as, and is designed to be mounted on top of, a standard 4x4 junction box, replacing the junction box's cover plate.

Before mounting the SSB-UI1 to the junction box, verify all wiring is correct, making sure that all screw terminals are sufficiently tightened and all terminal blocks are securely seated.

With the wires attached to the device, loosen the screws on the 4x4 junction box slightly. The screws should be loose enough to provide room to slide the SSB-UI1 onto the screws, but not so loose that they can fall out. Align the channels on the back corners of the SSB-UI1 with the screws on the junction box and slide the unit downward onto the screws as shown in Figure 5-16. Tighten the screws through the holes in the front of the SSB-UI1 to secure the device to the junction box.

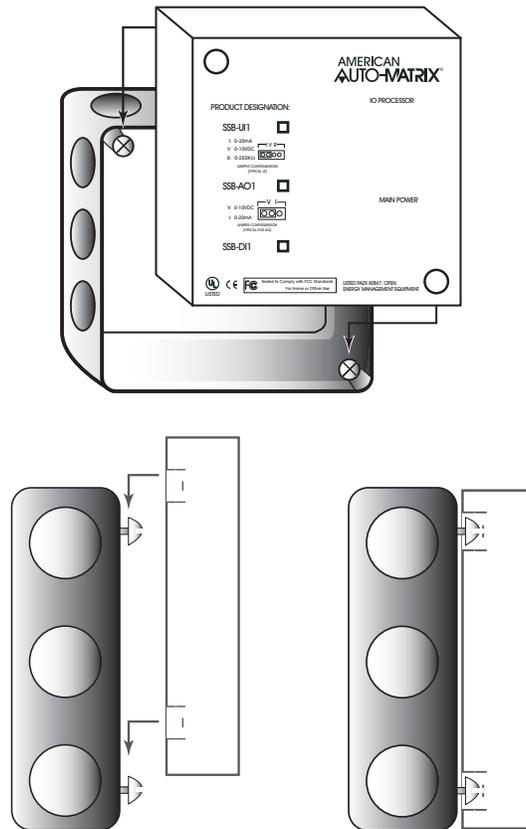


Figure 5-16: Mounting the SSB-UI1 to a 4x4 junction box

#### 5.7.4 STATUS INDICATOR LED

The SSB-UI1 has an IO Processor indicator LED which provides feedback as to the device's current operational status. The IO Processor indicator LED is located on the front of the SSB-UI1 (the side that faces out when installed) as shown in Figure 5-17. This allows status diagnostics to be performed without having to remove the SSB-UI1.

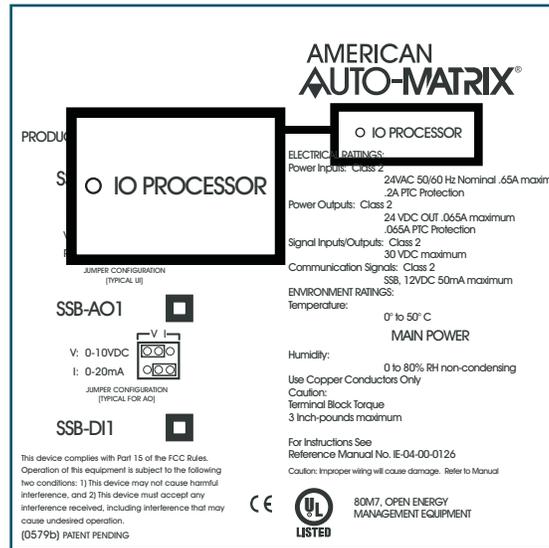


Figure 5-17: Location of the IO Processor Indicator LED on the SSB-UI1

The status indicator LED shows one of four different states: powered but not enumerated, enumerated but not configured, configured, and “identify”. The different states are indicated by the rate at which the LED blinks. The LED blinking quickly, approximately three to four times per second, indicates that the unit is powered but has not yet been enumerated by the controller. This is useful for identifying units that are correctly wired but not configured. When the device is enumerated but not configured, the blink rate will slow to approximately twice a second.

Once the device has been configured, the blink rate will slow down to approximately one blink per second. This will be the normal state of the device when it is correctly wired, powered, enumerated, and configured in the controller.

When the controller is set to “Identify” in the Configure Function and Configure Device properties, the LED on the SSB-UI1 will be blink three times in quick succession and then pause before repeating the three blinks again. This is especially useful for quickly identifying an individual device in the field when troubleshooting the STATbus.

### 5.7.5 SSB-UI CONFIGURATION TABLE

Table 5-3: SSB-UI Configuration Table

I/O Quantity	(I#) Index Number
1 Universal Input	1

## 5.8 SSB-AO1

### 5.8.1 FEATURES

The SSB-AO1, shown in Figure 5-18, is a STATbus device which provides a single remote analog output to the GPC. This output can be configured as either a 0-10 VDC or 0-20 mA output via a user-selectable jumper.



**PRODUCT DESIGNATION:**

**SSB-UI1** 

I: 0-20mA   
V: 0-10VDC   
R: 0-250K $\Omega$  

JUMPER CONFIGURATION  
(TYPICAL UI)

**SSB-AO1** 

V: 0-10VDC   
I: 0-20mA 

JUMPER CONFIGURATION  
(TYPICAL FOR AO)

**SSB-DI1** 

This device complies with Part 15 of the FCC Rules.  
Operation of this equipment is subject to the following  
two conditions: 1) This device may not cause harmful  
interference, and 2) This device must accept any  
interference received, including interference that may  
cause undesired operation.  
(0579b) PATENT PENDING

**IO PROCESSOR**

**ELECTRICAL RATINGS:**  
Power Inputs: Class 2  
24VAC 50/60 Hz Nominal .65A maximum  
.2A PTC Protection

Power Outputs: Class 2  
24 VDC OUT .065A maximum  
.065A PTC Protection

Signal Inputs/Outputs: Class 2  
30 VDC maximum

Communication Signals: Class 2  
SSB, 12VDC 50mA maximum

**ENVIRONMENT RATINGS:**  
Temperature:  
0° to 50° C

**MAIN POWER**

Humidity:  
0 to 80% RH non-condensing

Use Copper Conductors Only  
Caution:  
Terminal Block Torque  
3 Inch-pounds maximum

For Instructions See  
Reference Manual No. IE-04-00-0125  
Caution: Improper wiring will cause damage. Refer to Manual



80M7, OPEN ENERGY  
MANAGEMENT EQUIPMENT

Figure 5-18: The SSB-AO1

### 5.8.2 WIRING/CONFIGURATION

Connections are made to the SSB-AO1 via the terminal blocks shown in Figure 5-19. The terminal blocks are located on the side of the unit which faces the junction box. There are two terminal blocks, the first is for connections to the STATbus network and power and the second is for the connection to the output device.

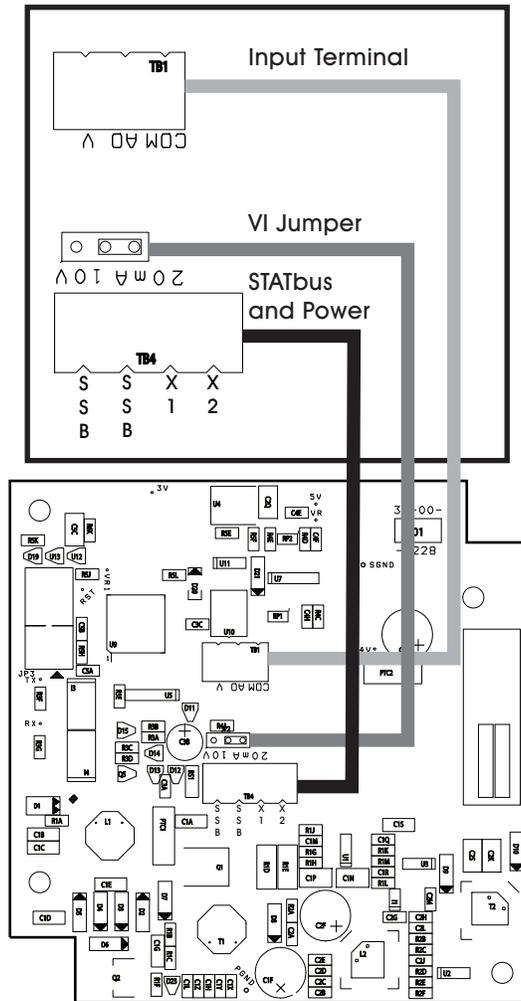


Figure 5-19:SSB-AO1 Terminal Block Locations

Each SSB-AO1 has a VI jumper identical to the ones found on the GPC. This is used to select the output range of the SSB-AO1. The output can be configured for 0-10 VDC or 0-20 mA operation, using the jumper positions shown in Figure 5-20a and b respectively.

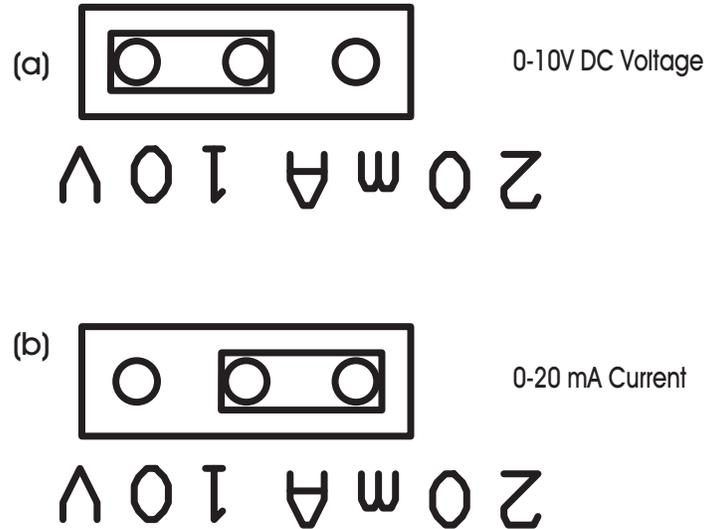


Figure 5-20: SSB-AO1 VI Jumper Positions

#### 5.8.2.1 OUTPUT WIRING

When using devices which do not require power, either because they do not require a power supply or because they have a dedicated external power supply, the SSB-AO1 is wired as shown in Figure 5-21. The signal wire should be connected to the AO terminal on the SSB-AO1 and the common wire should be connected to the COM terminal. For a 0-10 V device, the VI jumper should be set to the “V” position. If the device operated on a 0-20 mA signal, you would instead set the VI jumper to the “I” position.

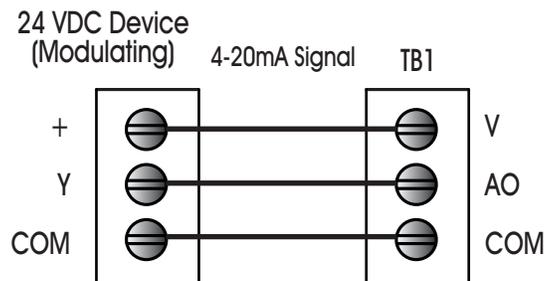


Figure 5-21: Wiring the SSB-AO1 to a Device That Does Not Require Power

## 5.8.2.2 POWERED DEVICES

When the output device requires power, the SSB-AO1 would be connected as shown in Figure 5-22. The + or power wire from the sensor should be connected to the V terminal on the SSB-AO1. The Y wire should be connected to the AO terminal and the common wire should be connected to the COM terminal. For a 0-10 V device, the VI jumper should be set to the "V" position. If the device operated on a 0-20 mA signal, you would instead set the VI jumper to the "I" position.

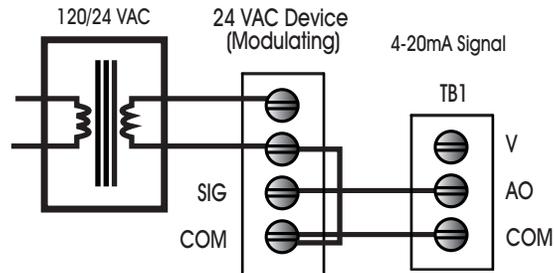


Figure 5-22: Wiring the SSB-AO1 to a Device That Requires Power

## 5.8.3 MOUNTING THE SSB-AO1

The SSB-AO1 is designed to be mounted on top of, a standard 4x4 junction box, replacing the junction box's cover plate.

Before mounting the SSB-AO1 to the junction box, verify all wiring is correct, making sure that all screw terminals are sufficiently tightened and all terminal blocks are securely seated.

With the wires attached to the device, loosen the screws on the 4x4 junction box slightly. The screws should be loose enough to provide room to slide the SSB-AO1 onto the screws, but not so loose that they can fall out. Align the channels on the back corners of the SSB-AO1 with the screws on the junction box and slide the unit downward onto the screws as shown in Figure 5-23. Tighten the screws through the holes in the front of the SSB-AO1 to secure the device to the junction box.

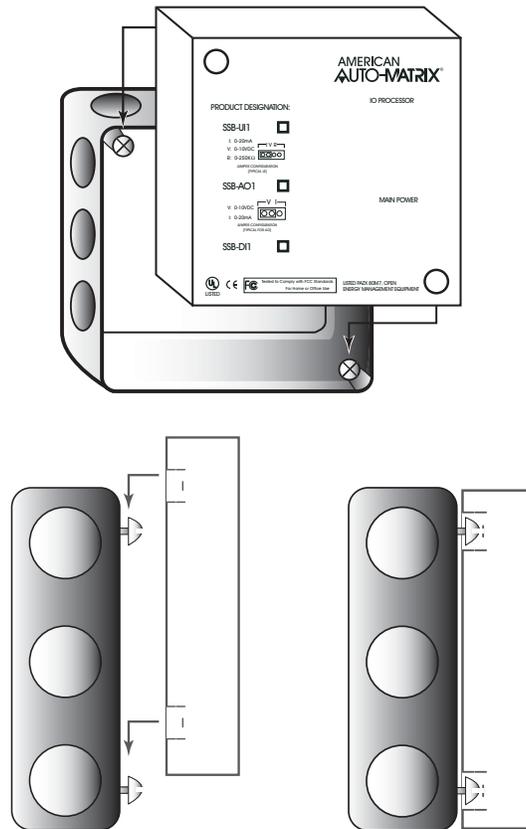


Figure 5-23: Mounting the SSB-AO1 to a 4x4 junction box

#### 5.8.4 STATUS INDICATOR LED

The SSB-AO1 has an IO Processor indicator LED which provides feedback as to the device's current operational status. The IO Processor indicator LED is located on the front of the SSB-AO1 (the side that faces out when installed) as shown in Figure 5-24. This allows status diagnostics to be performed without having to remove the SSB-AO1.

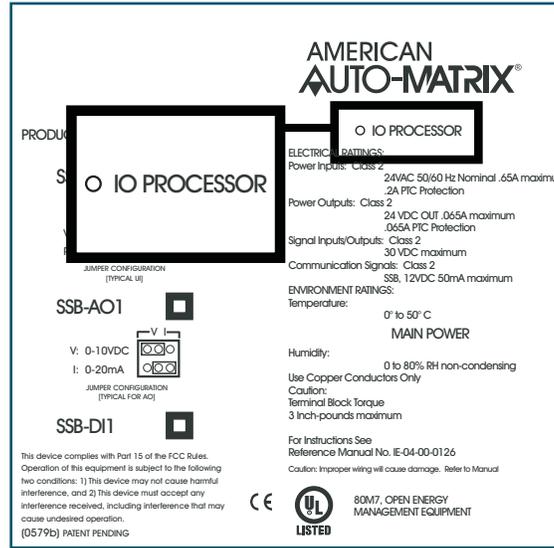


Figure 5-24: Location of the IO Processor Indicator LED on the SSB-AO1

The status indicator LED shows one of four different states: powered but not enumerated, enumerated but not configured, configured, and “identify”. The different states are indicated by the rate at which the LED blinks. The LED blinking quickly, approximately three to four times per second, indicates that the unit is powered but has not yet been enumerated by the controller.

Once the device has been enumerated, the blink rate will slow down to approximately one blink per second. This will be the normal state of the device when it is correctly wired, powered, enumerated, and configured in the controller.

When the controller is set to “Identify” in the Configure Function and Configure Device properties, the LED on the SSB-AO1 will be blink three times in quick succession and then pause before repeating the three blinks again. This is especially useful for quickly identifying an individual device in the field when troubleshooting the STATbus.

### 5.8.5 SSB-AO CONFIGURATION TABLE

Table 5-4: SSB-FI Configuration Table

I/O Quantity	(O#) Index Number
1 Analog Output	1

## 5.9 SSB-DI1

### 5.9.1 FEATURES

The SSB-DI1, shown in Figure 5-25, is a STATbus module which provides a single remote digital input to the GPC. This input is a wet contact that is capable of pulse counting.

AMERICAN  
AUTO-MATRIX®

**PRODUCT DESIGNATION:**

**SSB-UI1** 

I: 0-20mA   
V: 0-10VDC   
R: 0-250K $\Omega$  

JUMPER CONFIGURATION  
(TYPICAL UI)

**SSB-AO1** 

V: 0-10VDC   
I: 0-20mA 

JUMPER CONFIGURATION  
(TYPICAL AO)

**SSB-DI1** 

This device complies with Part 15 of the FCC Rules.  
Operation of this equipment is subject to the following  
two conditions: 1) This device may not cause harmful  
interference, and 2) This device must accept any  
interference received, including interference that may  
cause undesired operation.  
(0579b) PATENT PENDING

**IO PROCESSOR**

**ELECTRICAL RATINGS:**  
Power Inputs: Class 2  
24VAC 50/60 Hz Nominal .65A maximum  
.2A PTC Protection

Power Outputs: Class 2  
24 VDC OUT .065A maximum  
.05A PTC Protection

Signal Inputs/Outputs: Class 2  
30 VDC maximum

Communication Signals: Class 2  
SSB, 12VDC 50mA maximum

**ENVIRONMENT RATINGS:**  
Temperature:  
0° to 50° C

**MAIN POWER**

Humidity:  
0 to 80% RH non-condensing

Use Copper Conductors Only

**Caution:**  
Terminal Block Torque  
3 Inch-pounds maximum

For Instructions See  
Reference Manual No. IE-04-00-0126

Caution: Improper wiring will cause damage. Refer to Manual

CE

  
**LISTED**

80M7, OPEN ENERGY  
MANAGEMENT EQUIPMENT

Figure 5-25: The SSB-DI1

### 5.9.2 WIRING/CONFIGURATION

Connections are made to the SSB-DI1 via the terminal blocks shown in Figure 5-26. The terminal blocks are located on the side of the unit which faces the junction box. There are two blocks, the first for connections to the STATbus network and power, and the second for the connection to the input device.

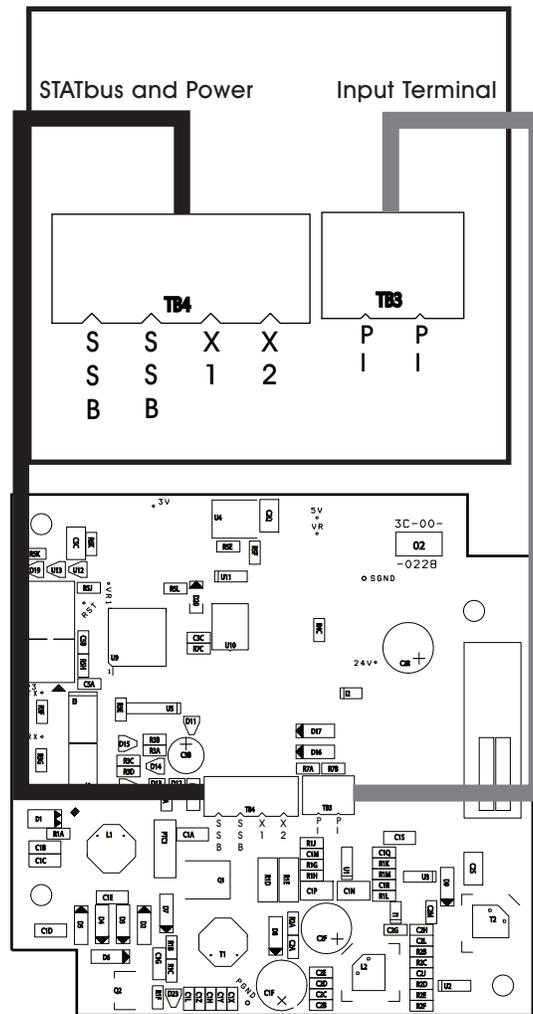


Figure 5-26:SSB-DI1 Terminal Block Locations

When the SSB-DI1 is configured to operate with an internally powered input, it would be wired as shown in Figure 5-27. The two wires from the input should be connected to the to PI terminals on the SSB-DI1.

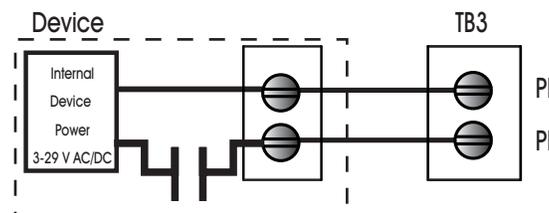


Figure 5-27: Wiring the SSB-DI1 for use with an Internally Powered Input

If the SSB-DI1 is to be used with an externally powered input, the wiring must include a source of power for the input device. As shown in Figure 5-28, the X1 and X2 terminals are connected to one of side of the sensor and one of the PI terminal. The other side of the sensor is wired to the remaining PI terminal.

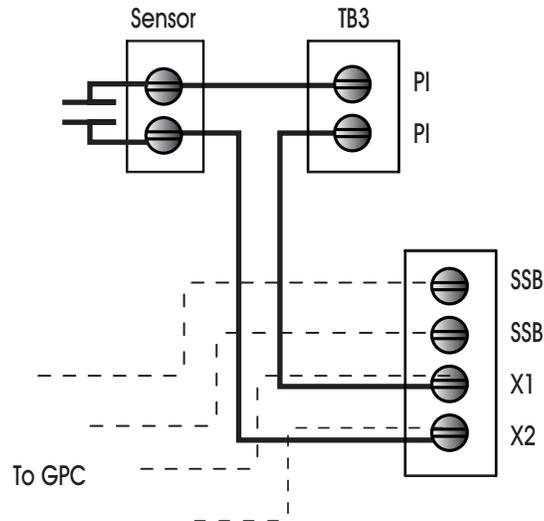


Figure 5-28: Wiring the SSB-DI1 for use with an Externally Powered Input

### 5.9.3 MOUNTING THE SSB-DI1

The SSB-DI1 has the same footprint and is designed to be mounted on top of a standard 4x4 junction box, replacing the junction box's cover plate.

Before mounting the SSB-DI1 to the junction box, verify all wiring is correct, making sure that all screw terminals are sufficiently tightened and all terminal blocks are securely seated.

With the wires attached to the device, loosen the screws on the 4x4 junction box slightly. The screws should be loose enough to provide room to slide the SSB-DI1 onto the screws, but not so loose that they can fall out. Align the channels on the back corners of the SSB-DI1 with the screws on the junction box and slide the unit downward onto the screws as shown in Figure 5-29. Tighten the screws through the holes in the front of the SSB-DI1 to secure the device to the junction box.

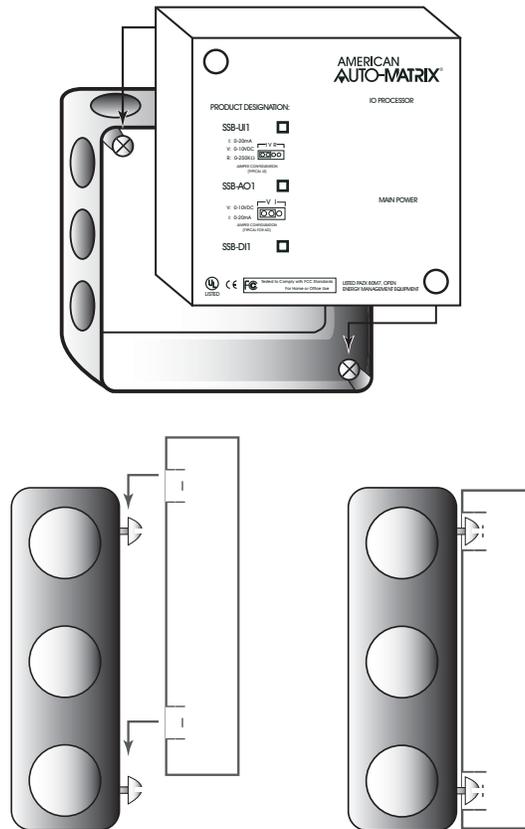


Figure 5-29: Mounting the SSB-DI1 to a 4x4 junction box

#### 5.9.4 STATUS INDICATOR LED

The SSB-DI1 has an IO Processor indicator LED which provides feedback as to the device's current operational status. The IO Processor indicator LED is located on the front of the SSB-DI1 (the side that faces out when installed) as shown in Figure 5-30. This allows status diagnostics to be performed without having to remove the SSB-DI1.

#### 5.9.5 SSB-DI1 CONFIGURATION TABLE

Table 5-5: SSB-FI Configuration Table

I/O Quantity	(I#) Index Number
1 Digital Input	1

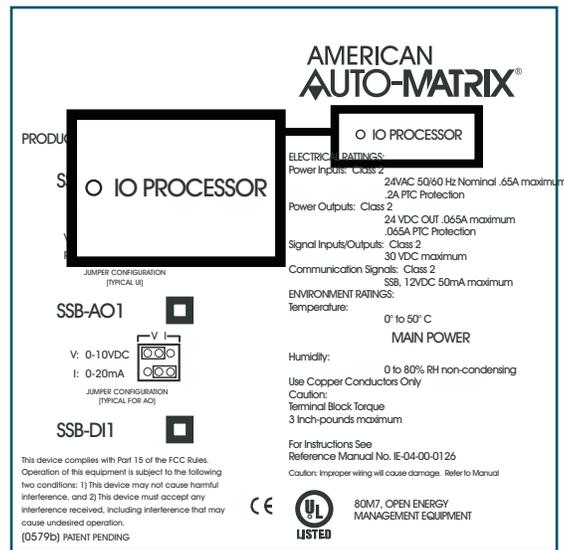


Figure 5-30: Location of the IO Processor Indicator LED on the SSB-DI1

The status indicator LED shows one of four different states: powered but not enumerated, enumerated but not configured, configured, and “identify”. The different states are indicated by the rate at which the LED blinks. The LED blinking quickly, approximately three to four times per second, indicates that the unit is powered but has not yet been enumerated by the controller. This is useful for identifying units that are correctly wired but not configured. When the device is enumerated but not configured, the blink rate will slow to approximately twice a second.

Once the device has been configured, the blink rate will slow down to approximately one blink per second. This will be the normal state of the device when it is correctly wired, powered, enumerated, and configured in the controller.

When the controller is set to “Identify” in the Configure Function and Configure Device properties, the LED on the SSB-ADI1 will be blink three times in quick succession and then pause before repeating the three blinks again. This is especially useful for quickly identifying an individual device in the field when troubleshooting the STATbus.

## 5.10 SSB-DO1

### 5.10.1 FEATURES

The SSB-DO1, shown in Figure 5-31, is a STATbus module which provides a single remote digital output to the GPC. The digital output on the SSB-DO1 is a relay capable of switching up to 250 VAC/DC at up to 10 A.

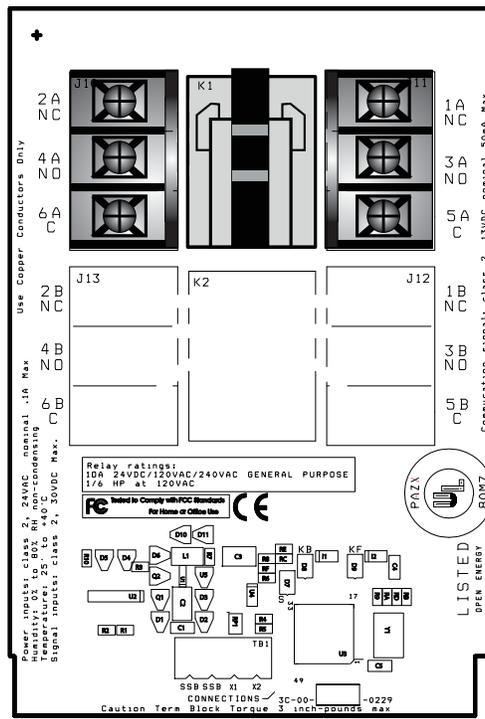


Figure 5-31: The SSB-DO1

### 5.10.2 MOUNTING THE SSB-DO1

The SSB-DO1 is intended to be installed near the equipment it is going to control. The module is designed to be mounted in the snap-in plastic track included with the module.

Before installing the module, the snap-in plastic track should be attached to the mounting surface using the screws provided. Once the snap-in plastic track has been firmly attached, the SSB-DO1 should be installed by first inserting one edge of the module in the channel on the inside of one side of the rail and then pressing the module so that it snaps into the other side. This is shown in Figure 5-32.

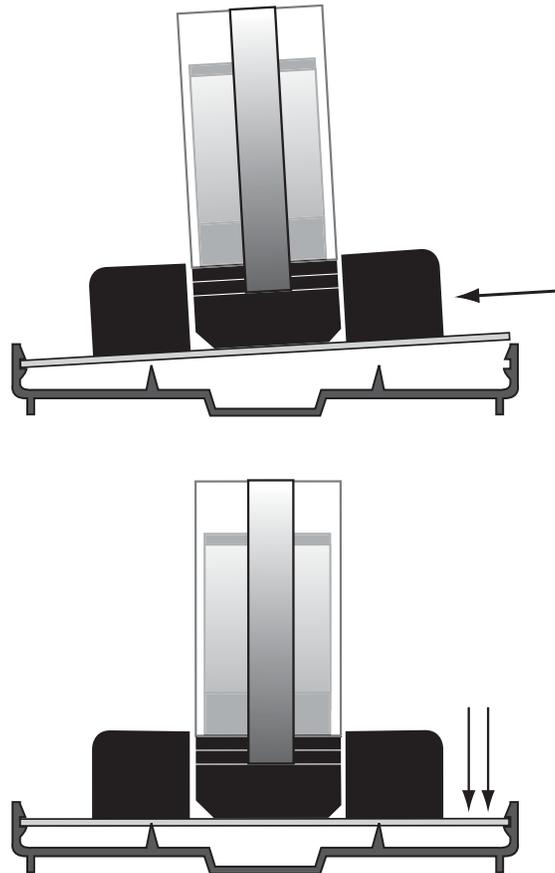


Figure 5-32: Attaching the SSB-DO1 to the Snap-in Plastic Track

### WARNING



**To avoid damage to the module, you should not attempt to slide the module into the snap-in plastic track.**

Once the module has been properly mounted, you may make the connections for STATbus and power.

## 5.10.3 WIRING/CONFIGURATION

### 5.10.3.1 POWER AND STATBUS CONNECTIONS

Connections for power and STATbus communications are made via terminal block TB1, shown in Figure 5-33. The two wire connection for STATbus communications should be connected to the two left most terminals labeled “SSB”. A source of 24VAC should be connected to the right two terminals labeled “X1” and “X2”. The easiest way to connect these terminals is to connect them to the AC OUT terminals on the same terminal block as the SSB connection on the controller. This allows you to simply run a 4-conductor

cable to connect both power and STATbus communications to the device without any additional wiring. Alternately, a dedicated external power supply may be provided for the module.

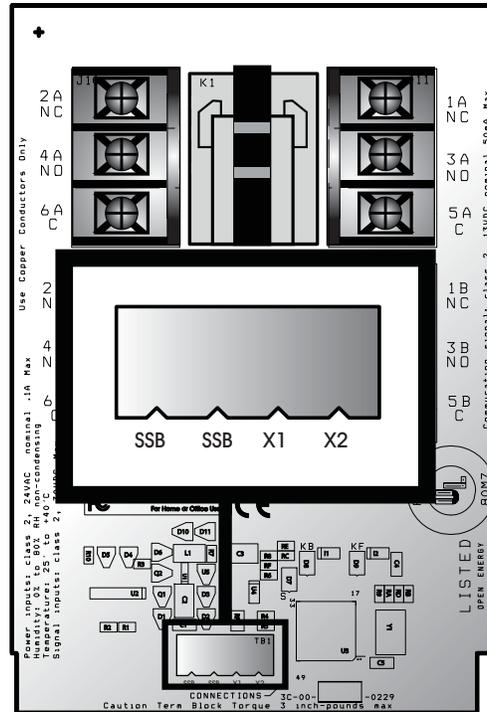


Figure 5-33: SSB and Power Connections on the SSB-DO1

### 5.10.3.2 RELAY CONNECTIONS

The SSB-DO1 has a single DPDT relay for its output. When the output is energized, the connection is made between the normally open terminal, labeled 3A and 4A, and the common terminals, labeled 5A and 6A. The normally open terminals are labeled “NO” below the terminal number and the common terminals are labeled “C”. When the output is not energized, connections will be made between the normally closed terminals, labeled 1A and 2A, and the common terminals. Like the normally open terminals, the normally closed terminals are labeled “NC” below the terminal number. The function of each of the relay terminals is summarized in Table 5-6:

Table 5-6: SSB-DO1 Relay Terminals

Terminal	Connection
1A	Normally Closed (NC)
2A	Normally Closed (NC)
3A	Normally Open (NO)
4A	Normally Open (NO)
5A	Common
6A	Common

## 5.10.4 SSB-DO1 CONFIGURATION TABLE

*Table 5-7: SSB-FI Configuration Table*

I/O Quantity	(O#) Index Number
1 Digital Output	1

## 5.11 SSB-DO1-I

### 5.11.1 FEATURES

The SSB-DO1-I, shown in Figure 5-34, is identical to the SSB-DO1 except that it includes a single dry contact, digital input. The digital output on the SSB-DO1-I is a relay capable of switching up to 250 VAC/DC at up to 10 A. The digital input is a dry contact which is intended for status monitoring of the output state of the relay. Connecting a wet contact to this input will result in damage to the SSB-DO1-I. This input is mapped to a Digital Input on the GPC but, unlike the digital input on the GPC, is not capable of performing pulse counting.

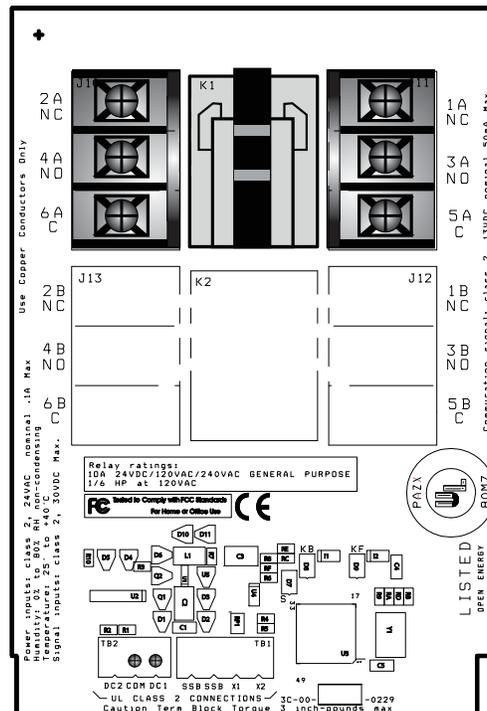


Figure 5-34: The SSB-DO1-I

### 5.11.2 MOUNTING THE SSB-DO1-I

The SSB-DO1-I is intended to be installed near the equipment it is going to control. The module is designed to be mounted in the snap-in plastic track included with the module.

Before installing the module, the snap-in plastic track should be attached to the mounting surface using the screws provided. Once the snap-in plastic track has been firmly attached, the SSB-DO1-I should be installed by first inserting one edge of the module in the channel on the inside of one side of the rail and then pressing the module so that it snaps into the other side. This is shown in Figure 5-35.

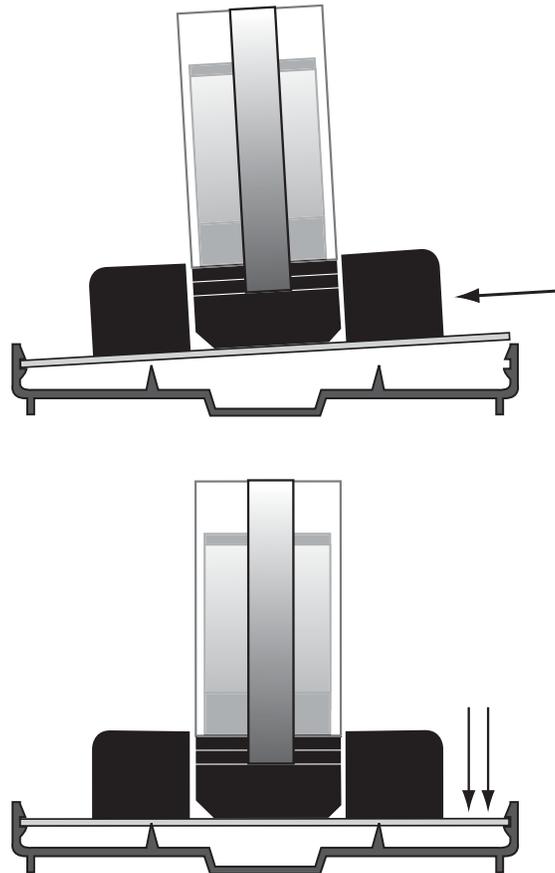


Figure 5-35: Attaching the SSB-DO1-I to the Snap-in Plastic Track

WARNING	
	<p><b>To avoid damage to the module, you should not attempt to slide the module into the snap-in plastic track.</b></p>

Once the module has been properly mounted, you may make the connections for STATbus and power.

### 5.11.3 WIRING/CONFIGURATION

#### 5.11.3.1 POWER AND STATBUS CONNECTIONS

Connections for power and STATbus communications are made via terminal block TB1, shown in Figure 5-36. The two wire connection for STATbus communications should be connected to the two left most terminals labeled “SSB”. A source of 24VAC should be connected to the right two terminals labeled “X1” and “X2”. The easiest way to connect these terminals is to connect them to the AC OUT terminals on the same terminal block as the SSB connection on the controller. This allows you to simply run a 4-conductor cable to connect both power and STATbus communications to the device without any additional wiring. Alternately, a dedicated external power supply may be provided for the module.

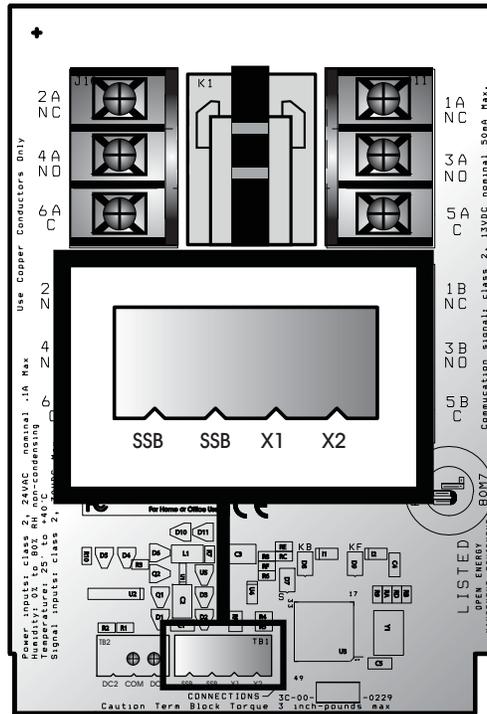


Figure 5-36: SSB and Power Connections on the SSB-DO1-I

5.11.3.2 RELAY CONNECTIONS

The SSB-DO1-I has a single DPDT relay for its output. When the output is energized, the connection is made between the normally open terminal, labeled 3A and 4A, and the common terminals, labeled 5A and 6A. The normally open terminals are labeled “NO” below the terminal number and the common terminals are labeled “C”. When the output is not energized, connections will be made between the normally closed terminals, labeled 1A and 2A, and the common terminals. Like the normally open terminals, the normally closed terminals are labeled “NC” below the terminal number. The function of each of the relay terminals is summarized in Table 5-8:.

Table 5-8: SSB-DO1-I Relay Terminals

Terminal	Connection
1A	Normally Closed (NC)
2A	Normally Closed (NC)
3A	Normally Open (NO)
4A	Normally Open (NO)
5A	Common
6A	Common

## 5.11.3.3 DRY CONTACT INPUT CONNECTION

The SSB-DO1-I has a single dry contact digital input intended for use as a status monitor for the relay on the module. Connecting a wet contact to this input will result in damage to the SSB-DO1-I. This input can only be assigned to a Digital Input in the controller. Unlike the on-board digital input on the GPC, the dry contact input on the SSB-DO1-I is not capable of performing pulse counting. Connections for the input are via the COM and DC1 terminals on terminal block TB2 as shown in Figure 5-37.

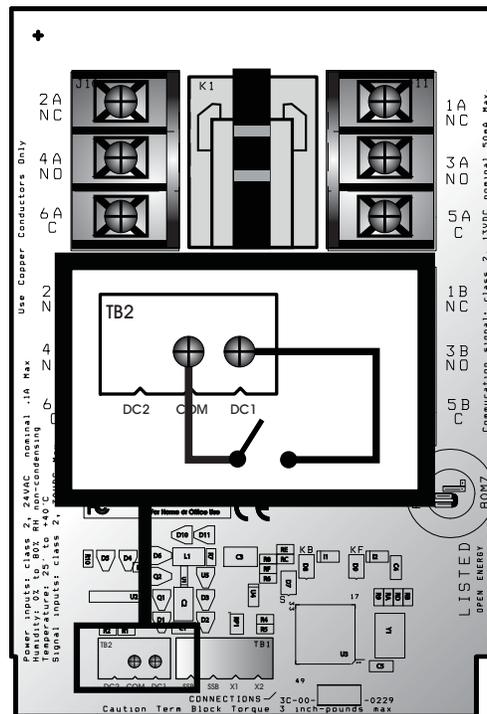


Figure 5-37: Connecting a Dry Contact Input to the SSB-DO1-I

## CAUTION



Assigning the SSB-DO1-I's **GID** number to the **GI**, and **I#** properties of an input only assigns the input on the module. The output will not be assigned unless the **GID** number and **O#** attribute are also assigned to a Digital Output in the controller.

## 5.11.4 SSB-DO1-I CONFIGURATION TABLE

*Table 5-9: SSB-DO1-I Configuration Table*

I/O Termination	(I# or O#) Index Number
Digital Output 1	1
Digital Input 1	1

## 5.12 SSB-DO2

### 5.12.1 FEATURES

The SSB-DO2, shown in Figure 5-38, is a STATbus module which provides two remote digital outputs to the GPC. The digital outputs on the SSB-DO2 are relays capable of switching up to 250 VAC/DC at up to 10 A each.

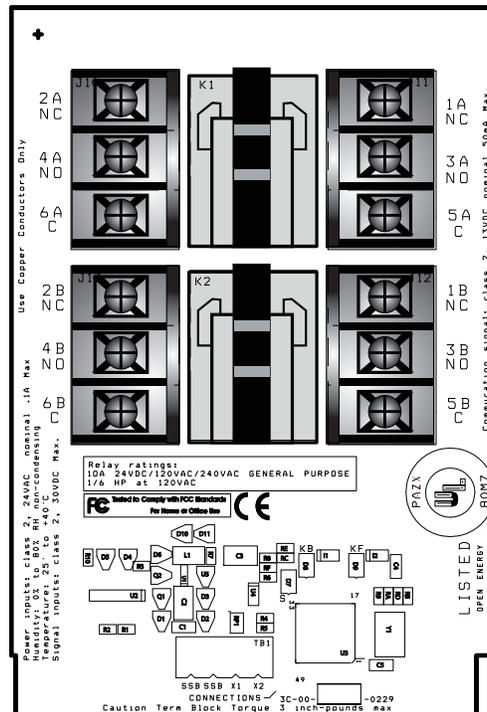


Figure 5-38: The SSB-DO2

### 5.12.2 MOUNTING THE SSB-DO2

The SSB-DO2 is intended to be installed near the equipment it is going to control. The module is designed to be mounted in the snap-in plastic track included with the module.

Before installing the module, the snap-in plastic track should be attached to the mounting surface using the screws provided. Once the snap-in plastic track has been firmly attached, the SSB-DO2 should be installed by first inserting one edge of the module in the channel on the inside of one side of the rail and then pressing the module so that it snaps into the other side. This is shown in Figure 5-39.

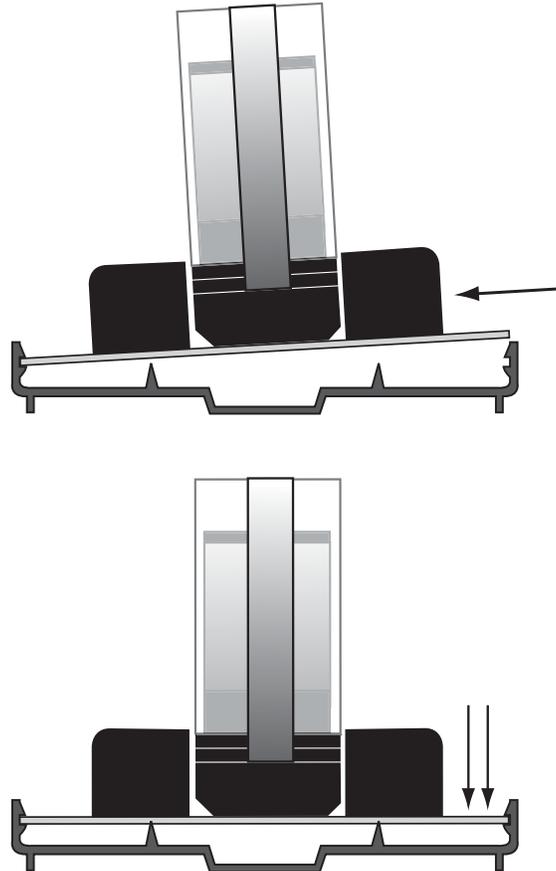


Figure 5-39: Attaching the SSB-DO2 to the Snap-in Plastic Track

WARNING	
	<p><b>To avoid damage to the module, you should not attempt to slide the module into the snap-in plastic track.</b></p>

Once the module has been properly mounted, you may make the connections for STATbus and power.

### 5.12.3 WIRING/CONFIGURATION

#### 5.12.3.1 POWER AND STATBUS CONNECTIONS

Connections for power and STATbus communications are made via terminal block TB1, shown in Figure 5-40. The two wire connection for STATbus communications should be connected to the two left most terminals labeled “SSB”. A source of 24VAC should be connected to the right two terminals labeled “X1” and “X2”. The easiest way to connect these terminals is to connect them to the AC OUT terminals on the same terminal block as the SSB connection on the controller. This allows you to simply run a 4-conductor cable to connect both power and STATbus communications to the device without any additional wiring. Alternately, a dedicated external power supply may be provided for the module.

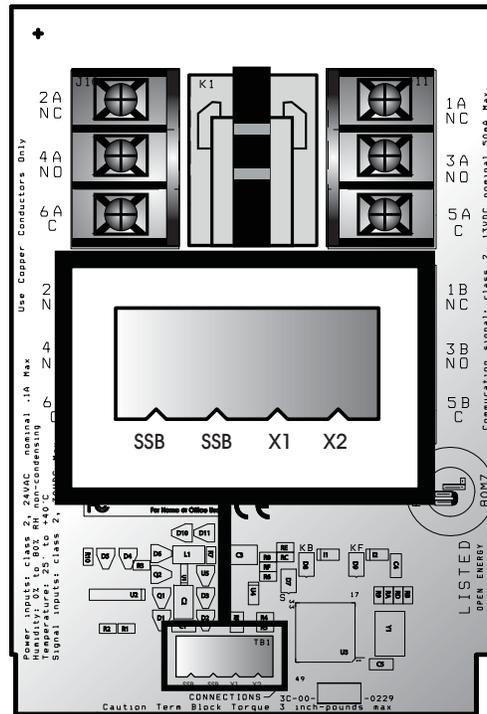


Figure 5-40: SSB and Power Connections on the SSB-DO2

### 5.12.3.2 RELAY CONNECTIONS

The SSB-DO2 has two (2) DPDT relays for its outputs. When output 1 is energized, the connection is made between the normally open terminal, labeled 3A and 4A, and the common terminals, labeled 5A and 6A. When the coil for output 2 is energized, the connection is made between the normally open terminal, labeled 3B and 4B, and the common terminals, labeled 5B and 6B. The normally open terminals are labeled “NO” below the terminal number and the common terminals are labeled “C”. When the output 1 is not energized connections will be made between the normally closed terminals, labeled 1A and 2A, and the common terminals. Similarly, when output 2 is no energized, connections will be made between the normally closed terminals, labeled 1B and 2B, and the common terminals. Like the normally open terminals, the normally closed terminals are labeled “NC” below the terminal number. The function of each of the relay terminals is summarized in Table 5-10:.

Table 5-10: SSB-DO2 Relay Terminals

Terminal	Connection
1A	Normally Closed (NC)
2A	Normally Closed (NC)
3A	Normally Open (NO)
4A	Normally Open (NO)
5A	Common

Table 5-10: SSB-DO2 Relay Terminals

Terminal	Connection
6A	Common
1B	Normally Closed (NC)
2B	Normally Closed (NC)
3B	Normally Open (NO)
4B	Normally Open (NO)
5B	Common
6B	Common

## 5.12.4 SSB-DO2 CONFIGURATION TABLE

Table 5-11: SSB-DO2 Configuration Table

I/O Termination	(I# or O#) Index Number
Digital Output 1	1
Digital Output 2	2

## 5.13 SSB-DO2-I

### 5.13.1 FEATURES

The SSB-DO2-I, shown in Figure 5-41, is identical to the SSB-DO2 except that it includes two dry contact, digital inputs for device status monitoring. The digital inputs are dry contacts which are intended for status monitoring of the output states of the relays. Connecting wet contacts to these inputs will result in damage to the SSB-DO2-I. These inputs are mapped to Digital Inputs on the GPC but, unlike the digital input on the GPC, are not capable of performing pulse counting.

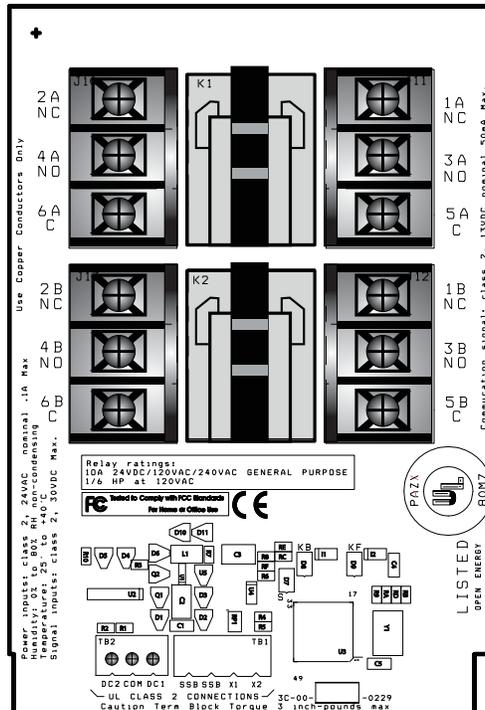


Figure 5-41: The SSB-DO2-I

### 5.13.2 MOUNTING THE SSB-DO2-I

The SSB-DO2-I is intended to be installed near the equipment it is going to control. The module is designed to be mounted in the snap-in plastic track included with the module.

Before installing the module, the snap-in plastic track should be attached to the mounting surface using the screws provided. Once the snap-in plastic track has been firmly attached, the SSB-DO2-I should be installed by first inserting one edge of the module in the channel on the inside of one side of the rail and then pressing the module so that it snaps into the other side. This is shown in Figure 5-42.

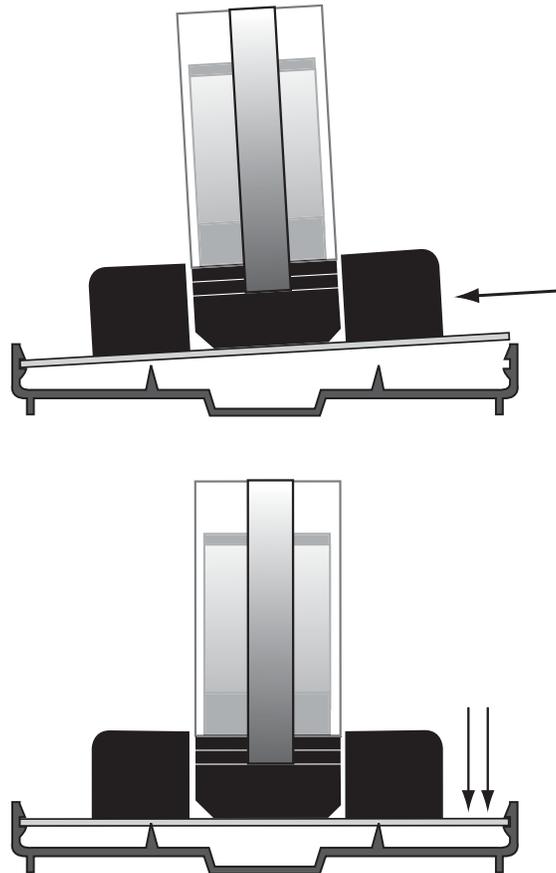


Figure 5-42: Attaching the SSB-DO2-I to the Snap-in Plastic Track

WARNING	
	<p><b>To avoid damage to the module, you should not attempt to slide the module into the snap-in plastic track.</b></p>

Once the module has been properly mounted, you may make the connections for STATbus and power.

### 5.13.3 WIRING/CONFIGURATION

#### 5.13.3.1 POWER AND STATBUS CONNECTIONS

Connections for power and STATbus communications are made via terminal block TB1, shown in Figure 5-43. The two wire connection for STATbus communications should be connected to the two left most terminals labeled “SSB”. A source of 24VAC should be connected to the right two terminals labeled “X1” and “X2”. The easiest way to connect these terminals is to connect them to the AC OUT terminals on the same terminal block as the SSB connection on the controller. This allows you to simply run a 4-conductor cable to connect both power and STATbus communications to the device without any additional wiring. Alternately, a dedicated external power supply may be provided for the module.

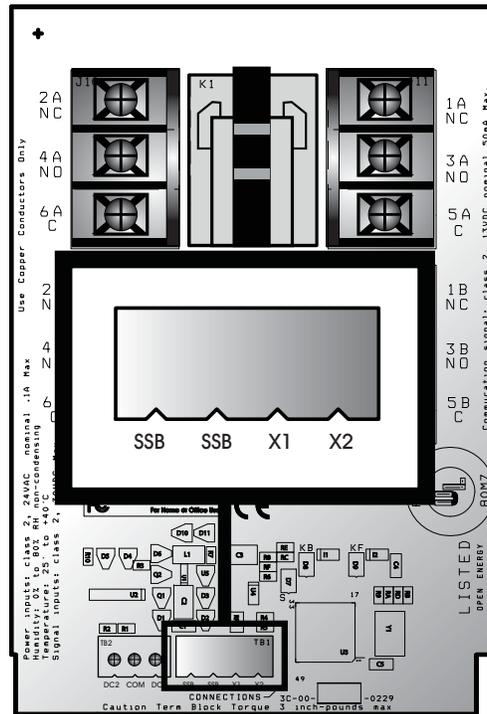


Figure 5-43: SSB and Power Connections on the SSB-DO2-I

### 5.13.3.2 RELAY CONNECTIONS

The SSB-DO2-I has two (2) DPDT relays for its outputs. When output 1 is energized, the connection is made between the normally open terminal, labeled 3A and 4A, and the common terminals, labeled 5A and 6A. When the coil for output 2 is energized, the connection is made between the normally open terminal, labeled 3B and 4B, and the common terminals, labeled 5B and 6B. The normally open terminals are labeled “NO” below the terminal number and the common terminals are labeled “C”. When the output 1 is not energized connections will be made between the normally closed terminals, labeled 1A and 2A, and the common terminals. Similarly, when output 2 is no energized, connections will be made between the normally closed terminals, labeled 1B and 2B, and the common terminals. The normally closed terminals are labeled “NC” below the terminal number. The function of each of the relay terminals is summarized in Table 5-12:.

Table 5-12: SSB-DO2-I Relay Terminals

Terminal	Connection
1A	Normally Closed (NC)
2A	Normally Closed (NC)
3A	Normally Open (NO)
4A	Normally Open (NO)
5A	Common

Table 5-12: SSB-DO2-I Relay Terminals

Terminal	Connection
6A	Common
1B	Normally Closed (NC)
2B	Normally Closed (NC)
3B	Normally Open (NO)
4B	Normally Open (NO)
5B	Common
6B	Common

5.13.3.3 DRY CONTACT INPUT CONNECTIONS

The SSB-DO2-I has two dry contact digital inputs intended for use as a status monitors for the relays on the module. Connecting a wet contact to these input will result in damage to the SSB-DO2-I. These inputs can only be assigned to Digital Inputs in the controller. Unlike the on-board digital input on the GPC, the dry contact inputs on the SSB-DO1-I are not capable of performing pulse counting. Connections for the inputs are made on terminal block TB2, with the first dry contact connected to the COM and DC1 terminals and the second dry contact connected to the COM and DC2 terminals as shown in Figure 5-44.

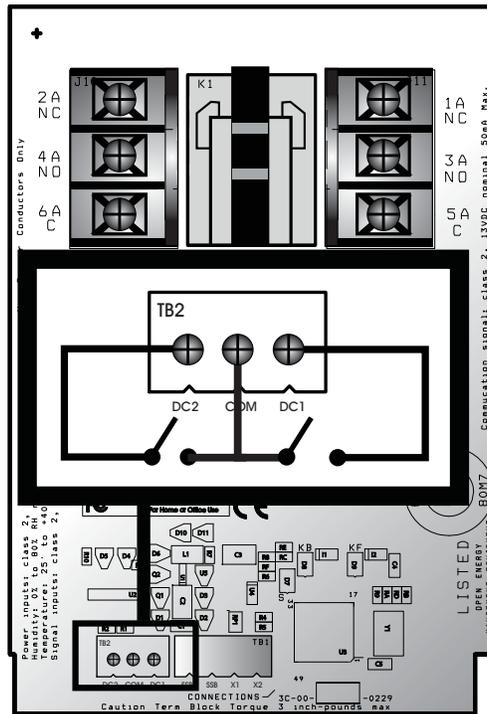


Figure 5-44: Connecting Dry Contact Inputs to the SSB-DO2-I

## 5.13.4 SSB-DO1-I CONFIGURATION TABLE

*Table 5-13: SSB-DO1-I Configuration Table*

I/O Termination	(I# or O#) Index Number
Digital Output 1	1
Digital Output 2	2
Digital Input 1	1
Digital Input 2	2

## 5.14 SSB-IOX FAMILY

The SSB-IOX Module Family are multi-I/O STATbus modules - providing multiple universal inputs, multiple analog outputs, and multiple digital outputs in contrast to singular SSB modules such as the SSB-UI1, SSB-AO1, and others.

SSB-IOX modules are available in four models, described below in Table 5-14.

*Table 5-14 SSB-IOX Family Devices*

Model	UIs	DIIs	AOs	DOs
SSB-IOX1-1	4	1	2	2
SSB-IOX1-2	8	-	-	-
SSB-IOX2-1	12	-	6	6
SBC-IOX2-2	12	-	-	-

## 5.15 SSB-IOX1-x

### 5.15.1 SSB-IOX1-1 FEATURES

The SSB-IOX1-1, shown in Figure 5-45, is a STATbus module based on the same hardware as the GPC2 controller. It provides four (4) universal inputs, one (1) pulse input, two (2) analog outputs, and two (2) digital outputs.

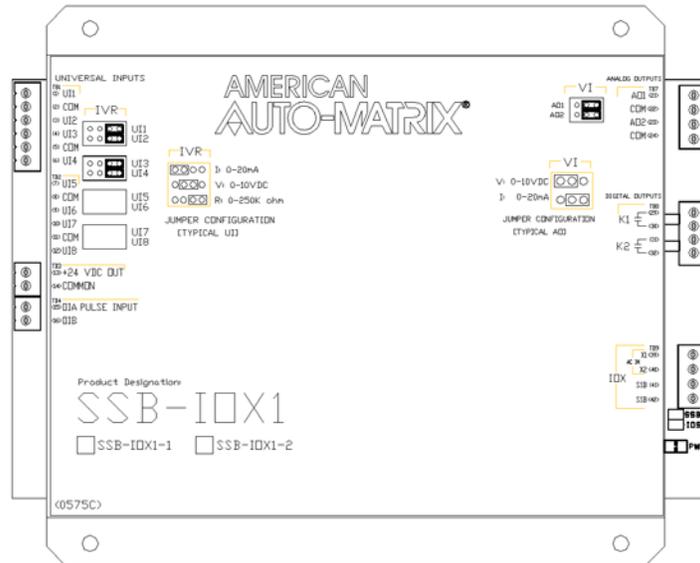


Figure 5-45 : The SSB-IOX1-1 Module

### 5.15.2 SSB-IOX1-2 FEATURES

The SSB-IOX1-2, shown in Table 5-46, is a STATbus module based on the same hardware as the GPC 2 controller. It provides eight (8) universal inputs.

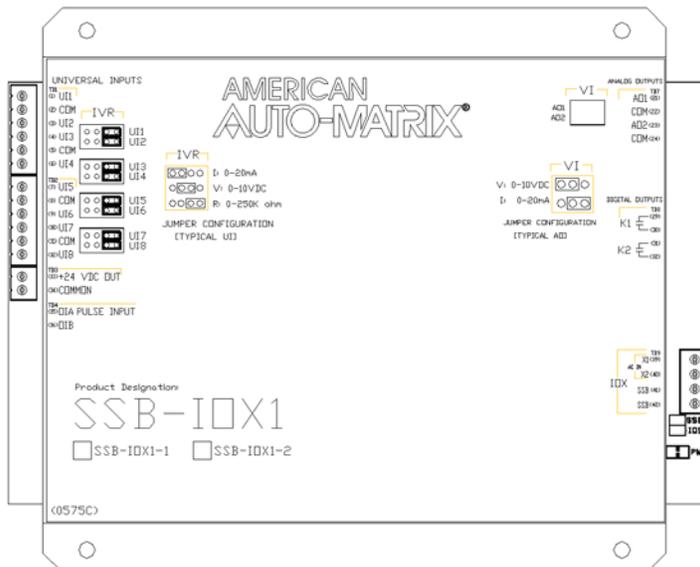


Figure 5-46 : The SSB-IOX1-2 Module

### 5.15.3 WIRING/CONFIGURATION

To properly configure the SSB-IOX1-X modules, you must connect the wires for network and power, connect any inputs and/or outputs, configure the IVR jumpers for Universal Inputs, and configure the VI jumpers for any Analog Outputs.

### 5.15.4 NETWORK & POWER

The SSB-IOX1-X must be connected to the STATbus network so that it may communicate. The network connection is made to terminal 41 and 42, labeled SSB, of terminal block TB9. The location of these terminals are shown in Figure 5-47. Power for the module is connected to terminals 39 and 40, labeled X1 and X2, on the same terminal block. This power may be provided from the AC Out terminals of the STATbus connection at the controller or a dedicated transformer may be connected.

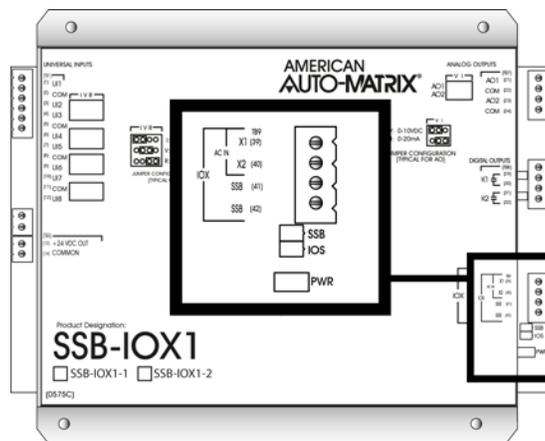


Figure 5-47: Location of the Network and Power Connections on the SSB-IOX1-1

### 5.15.5 UNIVERSAL INPUTS

To properly connect and configure the Universal Inputs on the SSB-IOX1-X, you must connect the sensor to the input and configure the IVR jumper to specify the type of sensor connected. The Universal Inputs are located in the upper left corner of the controller, as shown in Figure 5-48.

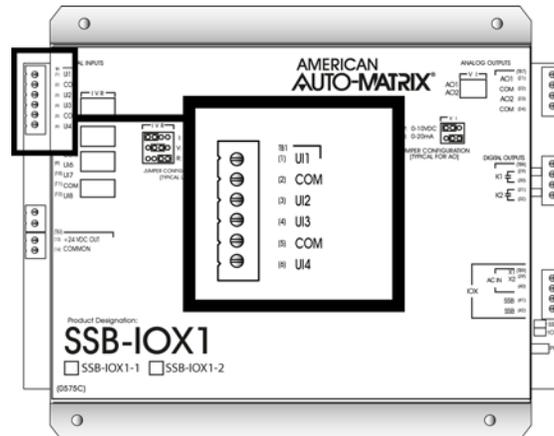


Figure 5-48: Location of the Universal Inputs on the SSB-IOX1-1 Module

To connect an input device to the SSB-IOX1-X, you must insert the leads from the sensor into the terminals for the desired input and the adjacent COM terminal. Two inputs share a single COM terminal, i.e. UIs 1 and 2 use the COM connection on terminal 2 while UIs 3 and 4 use the COM connection on terminal 5. Figure 5-49 shows how a thermistor would be connected to UI3.

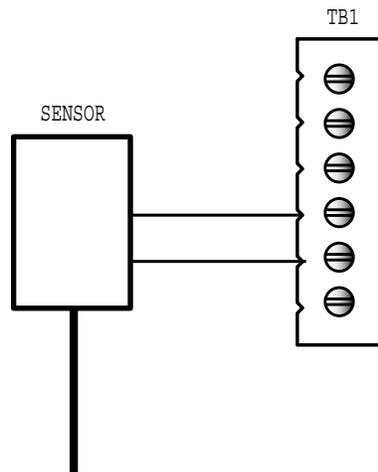


Figure 5-49: Connecting a Sensor to an Input on the SSB-IOX1-1

Each Universal Input on the SSB-IOX1-X has an IVR jumper, shown in Figure 5-50, associated with it which is used to select the type of input connected to the corresponding input.

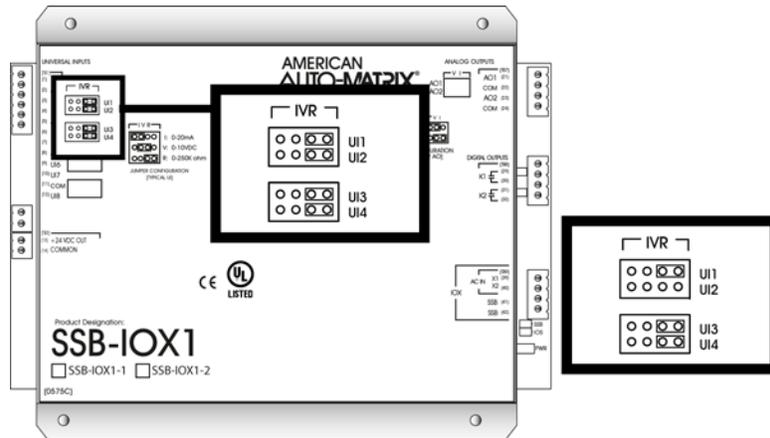


Figure 5-50: Location of the IVR Jumpers on the SSB-IOX1-1

Each input can be configured to read a 0-20 mA, 0-10 V or a 0-250 kΩ. The jumper settings corresponding to these options are shown in Figure 5-51a-c respectively.

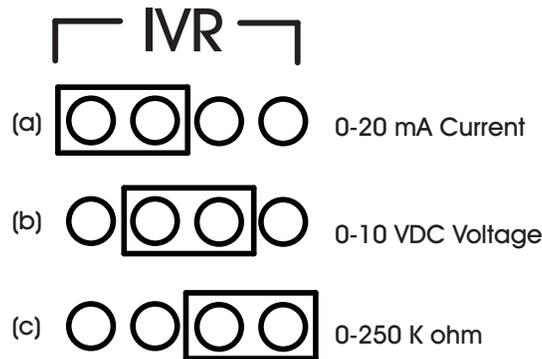


Figure 5-51: SSB-IOX1 IVR Jumper Positions

### 5.15.6 DIGITAL INPUTS

The SSB-IOX1-1 has a single Digital Input which is capable of performing high-speed pulse counting. The digital input is a wet contact input located on the left side of the module, as shown in Figure 5-52. There is a 24VDC power output connected to TB3 which is provided as a convenient way to power the wet contact connected to the input.

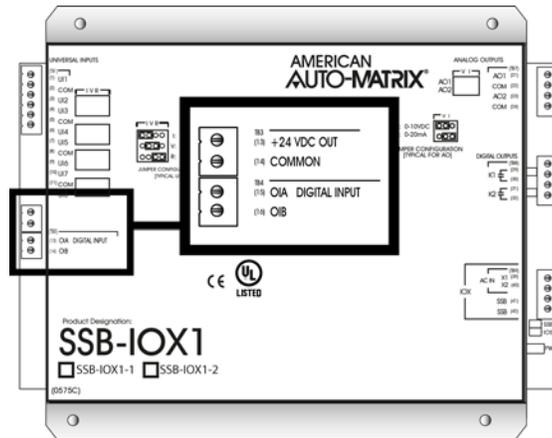


Figure 5-52: Location of the Digital Input and 24VDC Output on the SSB-IOX1-1

To connect a sensor to the pulse input on the SSB-IOX1-1, you must attach the leads from the sensor to the OIA and OIB terminals on terminal block TB4 as shown in Figure 5-53.

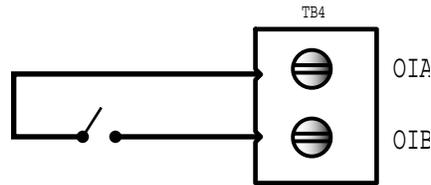


Figure 5-53: Connecting a Digital Input to the SSB-IOX1-1

### 5.15.7 ANALOG OUTPUTS

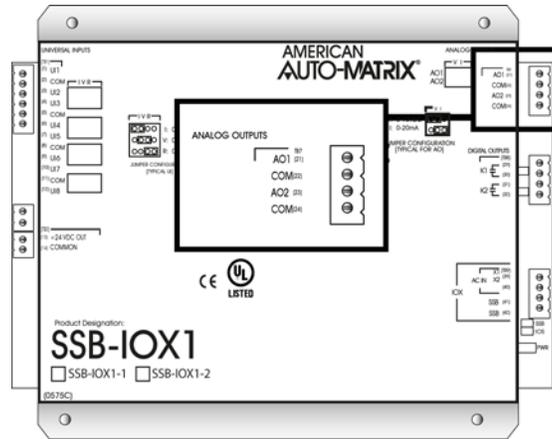


Figure 5-54: Location of the Analog Outputs on the SSB-IOX1-X

Analog outputs are connected to either terminals 21 (AO1) and 22 (COM) or 23 (AO2) and 24 (COM). A device connected to Analog Output 2 is shown in Figure 5-55.

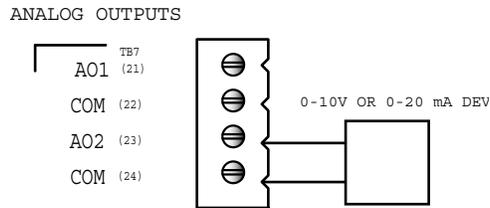


Figure 5-55: Connecting an Analog Output to the SSB-IOX1

For each Analog Output on the SSB-IOX1-1 there is a corresponding VI jumper used to select the output range for that output. These jumpers are located to the left of TB7. Each output can be configured for 0-10 VDC or 0-20 mA operation, using the jumper positions shown in Figure 5-56a and b respectively.

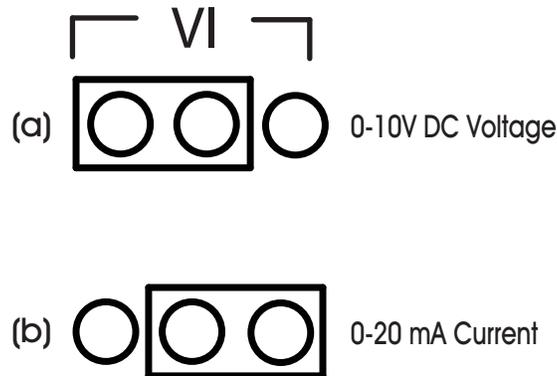


Figure 5-56: SSB-IOX1-1 VI Jumper Positions

### 5.15.8 DIGITAL OUTPUTS

The SSB-IOX1-1's two Digital Outputs are located on the right side of the controller as shown in Figure 5-57.

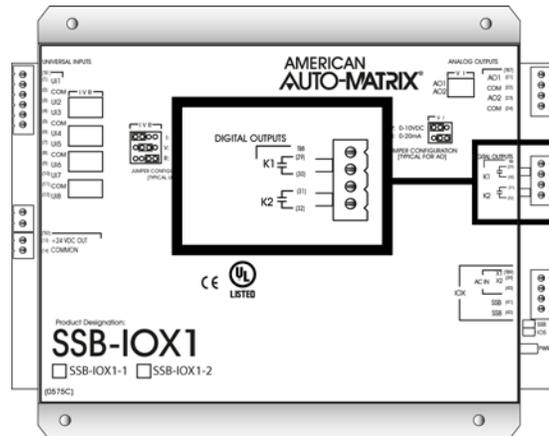


Figure 5-57: Location of the Digital Outputs on the SSB-IOX1-1

Output devices are connected using terminals 29 and 30 (labeled K1), for Digital Output 1, and terminals 31 and 32 (labeled K2), for Digital Output 2. Figure 5-58 shown a device connected to Digital Output 1.

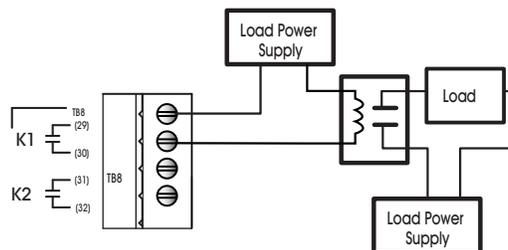


Figure 5-58: Connecting a Digital Output to the SSB-IOX1-1

### 5.15.9 MOUNTING THE SSB-IOX1-X

#### NOTE



The SSB-IOX1-1 and SSB-IOX1-2 are open type devices which, to meet UL specifications, are intended to be mounted on a panel that completes the ultimate enclosure.

The SSB-IOX1-X should be mounted to a site where the temperature is between 32° F and 122° F (0° C to 50° C) with a relative humidity of 0-80% non-condensing.

The mounting area should be flat and unobstructed by other equipment or machinery, free of moisture, and located away from potential leakage.

## NOTE



When installing the SSB-IOX1-1 and SSB-IOX1-2 make sure that there is sufficient room to allow insertion and removal of the terminal block plugs..

## 5.15.10 STATUS INDICATOR LED

The SSB-IOX1-X has an IOS indicator LED which provides feedback on the current operational status of the module's IO processor. The IOS LED is located on lower right side of the module as shown in Figure 5-59.

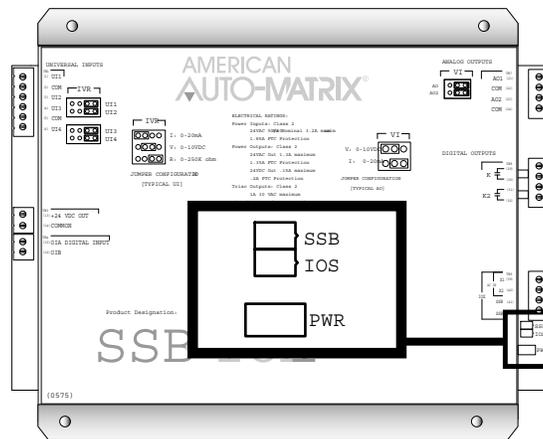


Figure 5-59: Location of the IOS Indicator LED on the SSB-IOX1-1

The IOS LED shows one of four different states: powered but not enumerated, enumerated but not configured, configured, and “identify”. The different states are indicated by the rate at which the LED blinks. The LED blinking quickly, approximately three to four times per second, indicates that the unit is powered but has not yet been enumerated by the controller. This is useful for identifying units that are correctly wired but not configured. When the device is enumerated but not configured, the blink rate will slow to approximately twice a second.

Once the device has been configured, the blink rate will slow down to approximately one blink per second. This will be the normal state of the device when it is correctly wired, powered, enumerated, and configured in the controller.

When the controller is set to “Identify” in the Configure Function and Configure Device properties, the IOS LED on the SSB-IOX1 will be blink three times in quick succession and then pause before repeating the three blinks again. This is especially useful for quickly identifying an individual device in the field when troubleshooting the STATbus.

## 5.15.11 SSB-IOX1-1 CONFIGURATION TABLE

Table 5-15: SSB-IOX1-1 Configuration Table

I/O Termination	(I# or O#) Index Number
Universal Input 1	1
Universal Input 2	2
Universal Input 3	3
Universal Input 4	4
Digital Input 1	1
Analog Output 1	1
Analog Output 2	2
Digital Output 1	1
Digital Output 2	2

## 5.15.11.1 SSB-IOX1-2 CONFIGURATION TABLE

Table 5-16 SSB-IOX1-2 Configuration Table

I/O Termination	(I# or O#) Index Number
Universal Input 1	1
Universal Input 2	2
Universal Input 3	3
Universal Input 4	4
Universal Input 5	5
Universal Input 6	6
Universal Input 7	7
Universal Input 8	8

## 5.16 SSB-IOX2-X

### 5.16.1 SSB-IOX2-1 FEATURES

The SSB-IOX2-1, shown in Figure 5-60, is a STATbus module based on the same hardware as the GPC1 controller. It provides twelve (12) universal inputs, six (6) analog outputs, and six (6) digital outputs.

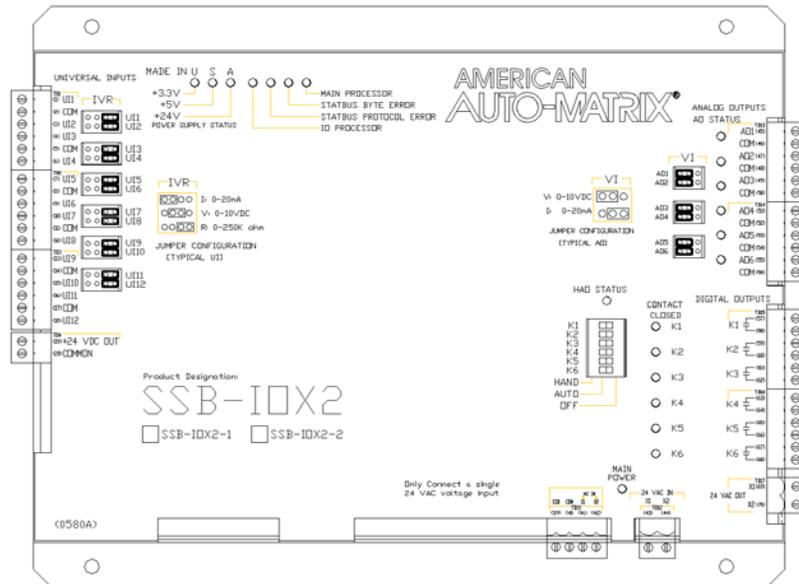


Figure 5-60 : The SSB-IOX2-1 Module

### 5.16.2 SSB-IOX2-2 MODULE

The SSB-IOX2-2, shown in Figure 5-61, is a STATbus module based on the same hardware as the GPC1 controller. It provides twelve (12) universal inputs.

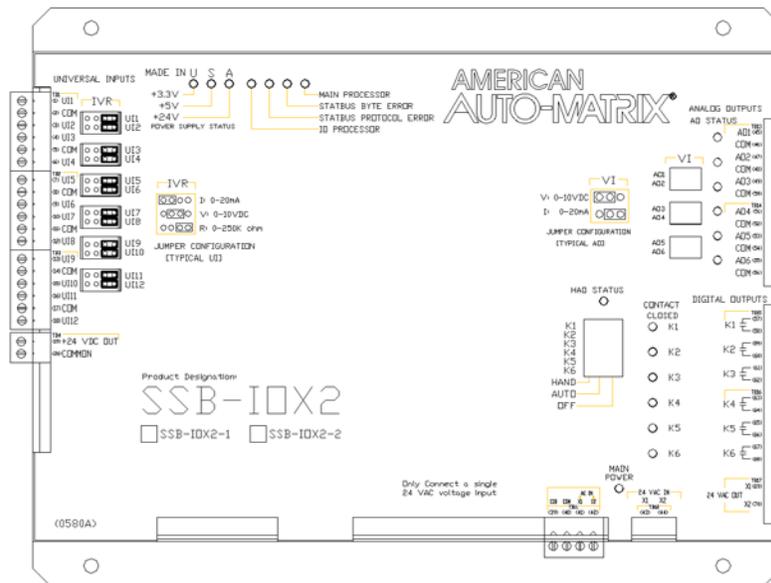


Figure 5-61 : The SSB-IOX2-2 Module

### 5.16.3 WIRING/CONFIGURATION

To properly configure the SSB-IOX2-X modules, you must connect the wires for network and power, connect any inputs and/or outputs, configure the IVR jumpers for Universal Inputs, and configure the VI jumpers for any Analog Outputs.

### 5.16.4 NETWORK & POWER

The SSB-IOX2-X must be connected to the STATbus network so that it may communicate. The network connection is made to terminal 39 and 40 labeled SSB, of terminal block TB11. The location of these terminals are shown in Figure 5-62. Power for the module is connected to terminals 41 and 42, labeled X1 and X2, on the same terminal block. This power may be provided from the AC Out terminals of the STATbus connection at the controller or a dedicated transformer may be connected. Alternatively, you may power the controller using a separate transformer using terminals 43 and 44 on TB12 if you intend to use the 24VAC out on TB17

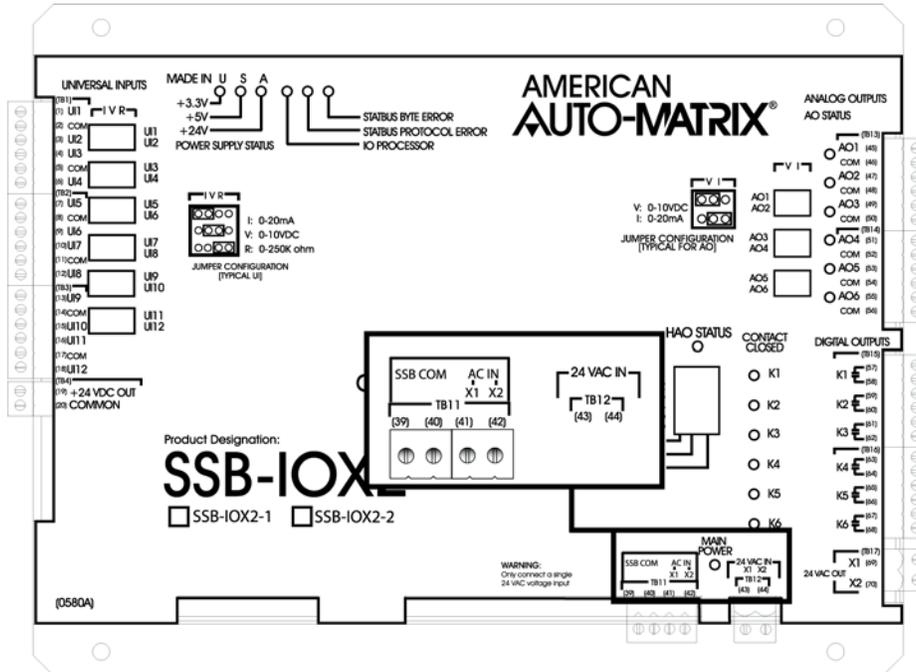


Figure 5-62: Location of the Network and Power Connections on the SSB-IOX2-1

### 5.16.5 UNIVERSAL INPUTS

To properly connect and configure the Universal Inputs on the SSB-IOX2-X, you must connect the sensor to the input and configure the IVR jumper to specify the type of sensor connected. The Universal Inputs are located in the upper left corner of the controller, as shown in Figure 5-63.

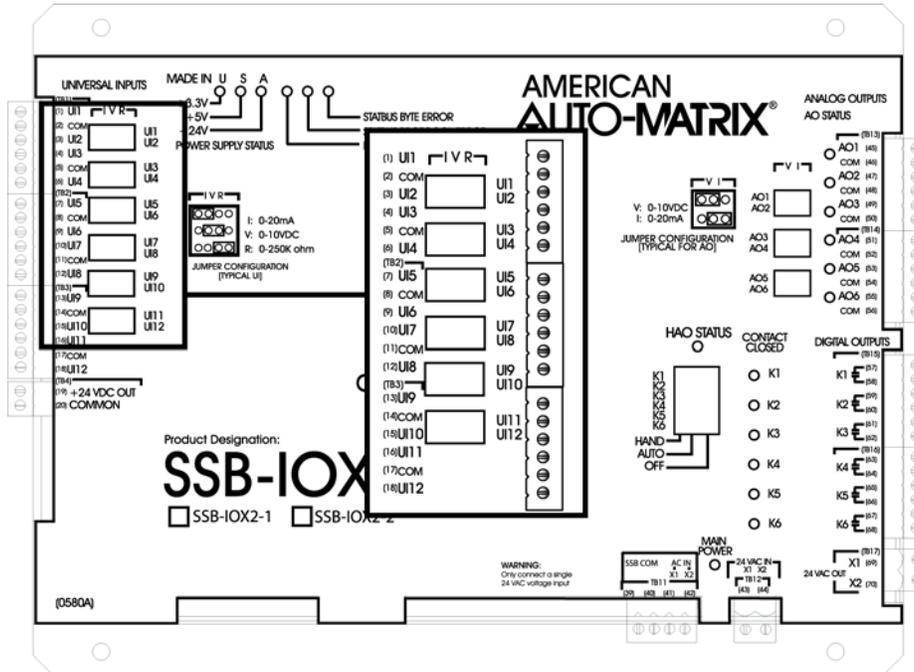


Figure 5-63: Location of the Universal Inputs on the SSB-IOX2-1 Module

To connect an input device to the SSB-IOX2-X, you must insert the leads from the sensor into the terminals for the desired input and the adjacent COM terminal. Two inputs share a single COM terminal, i.e. UIs 1 and 2 use the COM connection on terminal 2 while UIs 3 and 4 use the COM connection on terminal 5. Figure 5-49 shows how a thermistor would be connected to UI3.

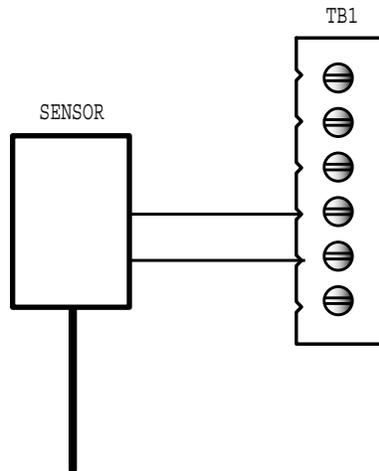


Figure 5-64: Connecting a Sensor to an Input on the SSB-IOX2-1

Each Universal Input on the SSB-IOX2-X has an IVR jumper, shown in Figure 5-65, associated with it which is used to select the type of input connected to the corresponding input.

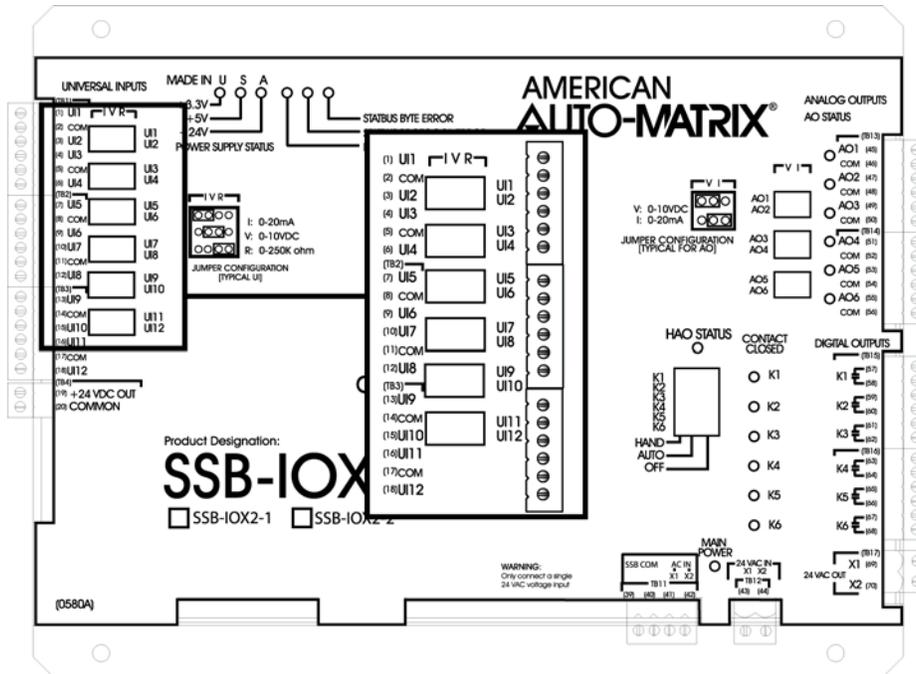


Figure 5-65: Location of the IVR Jumpers on the SSB-IOX2-1

Each input can be configured to read a 0-20 mA, 0-10 V or a 0-250 kΩ. The jumper settings corresponding to these options are shown in Figure 5-66 respectively.

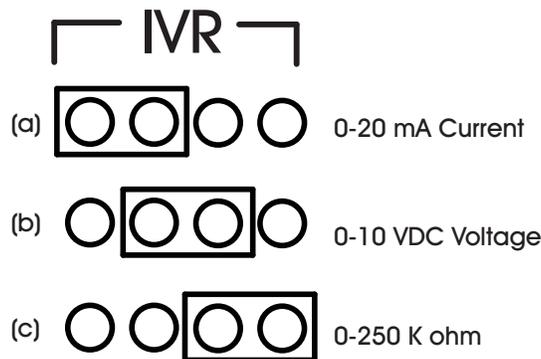


Figure 5-66: SSB-IOX2 IVR Jumper Positions

5.16.6 ANALOG OUTPUTS

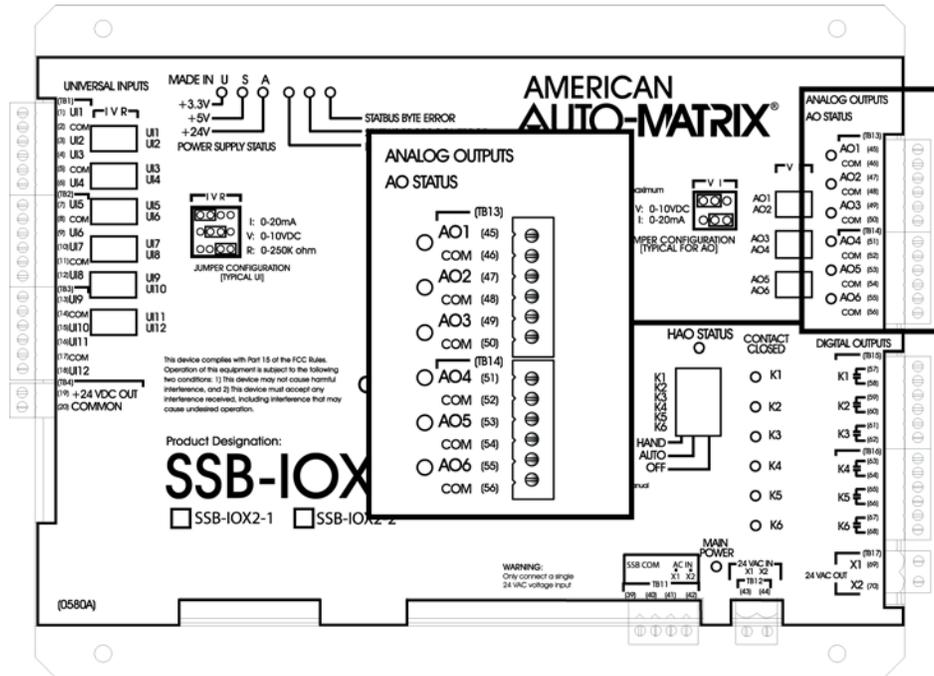


Figure 5-67: Location of the Analog Outputs on the SSB-IOX2

Analog outputs are connected to either terminals 45(AO1) and 46 (COM) or 47 (AO2) and 48 (COM). A device connected to Analog Output 2 is shown in Figure 5-55.

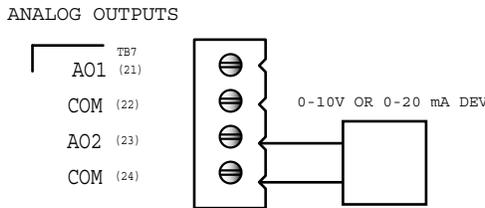


Figure 5-68: Connecting an Analog Output to the SSB-IOX1

For each Analog Output on the SSB-IOX2-1 there is a corresponding VI jumper used to select the output range for that output. These jumpers are located to the left of TB3 and TB4. Each output can be configured for 0-10 VDC or 0-20 mA operation, using the jumper positions shown in Figure 5-56a and b respectively.

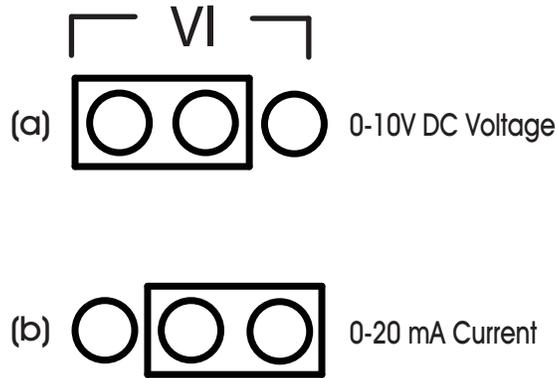


Figure 5-69: SSB-IOX1-1 VI Jumper Positions

### 5.16.7 DIGITAL OUTPUTS

The SSB-IOX2-1's six Digital Outputs are located on the right side of the controller as shown in Figure 5-70.

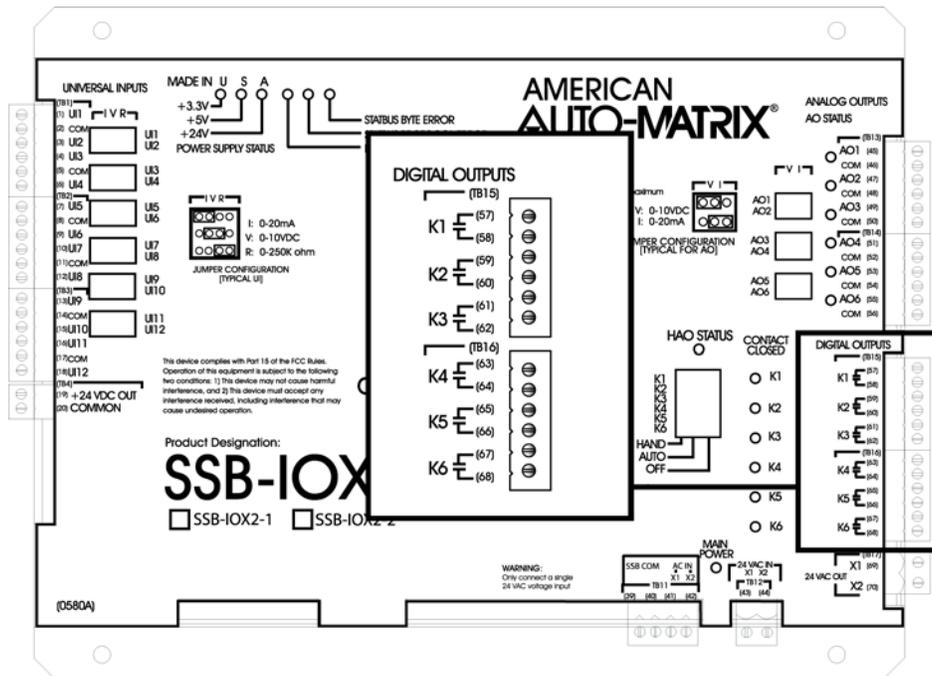


Figure 5-70: Location of the Digital Outputs on the SSB-IOX2-1

Output devices are connected using terminals 57 and 58 (labeled K1), for Digital Output 1, and terminals 60 and 61 (labeled K2), for Digital Output 2. Figure 5-58 shown a device connected to Digital Output 1.

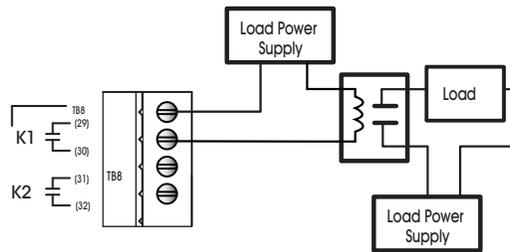


Figure 5-71: Connecting a Digital Output to the SSB-IOX2-1

### 5.16.8 MOUNTING THE SSB-IOX2-X

#### NOTE



The SSB-IOX2 is an open type device which, to meet UL specifications, is intended to be mounted on a panel that completes the ultimate enclosure.

The SSB-IOX2-X should be mounted to a site where the temperature is between 32° F and 122° F (0° C to 50° C) with a relative humidity of 0-80% non-condensing.

The mounting area should be flat and unobstructed by other equipment or machinery, free of moisture, and located away from potential leakage.

#### NOTE



When installing the SSB-IOX2-X make sure that there is sufficient room to allow insertion and removal of the terminal block plugs..

### 5.16.9 STATUS INDICATOR LED

The SSB-IOX2-X has an IOS indicator LED which provides feedback on the current operational status of the module's IO processor. The IOS LED is located on lower right side of the module as shown in Figure 5-72.

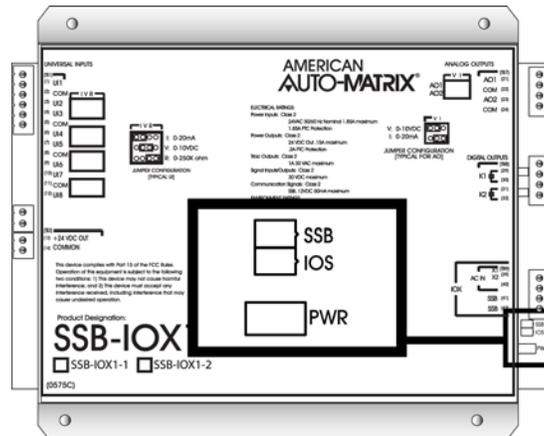


Figure 5-72: Location of the IOS Indicator LED on the SSB-IOX2-1

The IOS LED shows one of four different states: powered but not enumerated, enumerated but not configured, configured, and “identify”. The different states are indicated by the rate at which the LED blinks. The LED blinking quickly, approximately three to four times per second, indicates that the unit is powered but has not yet been enumerated by the controller. This is useful for identifying units that are correctly wired but not configured. When the device is enumerated but not configured, the blink rate will slow to approximately twice a second.

Once the device has been configured, the blink rate will slow down to approximately one blink per second. This will be the normal state of the device when it is correctly wired, powered, enumerated, and configured in the controller.

When the controller is set to “Identify” in the Configure Function and Configure Device properties, the IOS LED on the SSB-IOX2 will be blink three times in quick succession and then pause before repeating the three blinks again. This is especially useful for quickly identifying an individual device in the field when troubleshooting the STATbus.

### 5.16.10SSB-IOX2-1 CONFIGURATION TABLE

Table 5-17: SSB-IOX2-1 Configuration Table

I/O Termination	(I# or O#) Index Number
Universal Input 1	1
Universal Input 2	2
Universal Input 3	3
Universal Input 4	4
Universal Input 5	5
Universal Input 6	6

Table 5-17: SSB-IOX2-1 Configuration Table

I/O Termination	(I# or O#) Index Number
Universal Input 7	7
Universal Input 8	8
Universal Input 9	9
Universal Input 10	10
Universal Input 11	11
Universal Input 12	12
Analog Output 1	1
Analog Output 2	2
Analog Output 3	3
Analog Output 4	4
Analog Output 5	5
Analog Output 6	6
Digital Output 1	1
Digital Output 2	2
Digital Output 3	3
Digital Output 4	4
Digital Output 5	5
Digital Output 6	6

## 5.16.10.1 SSB-IOX2-2 CONFIGURATION TABLE

Table 5-18 SSB-IOX2-2 Configuration Table

I/O Termination	(I# or O#) Index Number
Universal Input 1	1
Universal Input 2	2
Universal Input 3	3
Universal Input 4	4
Universal Input 5	5

Table 5-18 SSB-IOX2-2 Configuration Table

I/O Termination	(I# or O#) Index Number
Universal Input 6	6
Universal Input 7	7
Universal Input 8	8
Universal Input 9	9
Universal Input 10	10
Universal Input 11	11
Universal Input 12	12

---

# SECTION 6: PROGRAMS AND FILES

*This section describes the use of SPL Programming with the GPC. A complete reference of how to use and write SPL programs is also reviewed in this section.*

## IN THIS SECTION

Overview .....	6-3
SPL Programming.....	6-4
Introduction to SPL .....	6-5
The Parts of SPL Programs .....	6-6
Program Names.....	6-7
The .SPL, .PLB and .LST Files .....	6-8
Attributes and Registers.....	6-9
Compiler Control Statements .....	6-10
Comments.....	6-12
Labels .....	6-13
Expressions .....	6-14
Program Statements Overview .....	6-16
Assignment Statements and Equates .....	6-18
Iteration, Branching and Subroutines.....	6-20
Program Delays .....	6-23
Execution Error Control.....	6-24
Debugging Statements.....	6-26
Program Control Attributes.....	6-27
Working with Channel Attributes .....	6-31



## 6.1 OVERVIEW

File objects are used to store compiled data for the GPC to use and execute. There are three specific applications where File objects are utilized.

1. In cases where an SPL Program has been loaded into a File object, a corresponding Program channel (F20x) invokes the file to execute the sequence and provides detailed feedback to users regarding the status of the program.
2. In cases where a site requires customization, LOGO files can be loaded into File objects which allowed connected SBC-STAT devices to display custom logos, list files, or other data.

The following information discusses each application.

## 6.2 SPL PROGRAMMING

The GPC supports the ability to accept line-by-line custom programming using the SPL Programming Language. Programs are written in the SPL Editor (located within SoloPro), and are then uploaded to program regions that exist within the GPC. As the program is loaded, a corresponding Program object provides status feedback regarding the program. The relationship between Program Region and Program channel is illustrated in the table below.

Table 6-1: File and Program Object Correlation

File Object	Program Channel
Region 1	F201
Region 2	F202
Region 3	F203
Region 4	F204
Region 5	F205
Region 6	F206
Region 7	F207
Region 8	F208

### 6.2.1 LOADING PROGRAMS INTO GPC

When a program is initially uploaded to the GPC using SoloPro's Upload/Download tab, the program will not execute until a user directly affects the current **(\$\$) Program Status**. To initially execute the program, set **(\$\$) Program Status** attribute to 6 (Load).

Once a program has been loaded, the program will continue to run. In the event that the GPC controller is reset (via warm-start, restart button, power cycle, etc.), the program will start automatically without user intervention.

### 6.3 INTRODUCTION TO SPL

The GPC Product Family leverages extension of the SAGE Programming Language to perform custom programming sequences. SPL support has been included in the GPC to allow users to create line-by-line, BASIC-like written sequences that cannot be carried out using the library of built-in logic objects.

The SPL Programming language support for GPC supports a large number of features, summarized below:

- . Unlimited number of program lines up to a maximum code block size of 8kb per program.
- . Up to 255 custom program properties (initialized using the ATTR statement)
- . Indirect references within the program up to 256 references
- . Floating point math and type coercion
- . Re-entrant subroutine calling ability for multiple programs
- . Six (6) Level expression stack to resolve nested expressions

## 6.4 THE PARTS OF SPL PROGRAMS

An SPL program consists of a collection of structures that are used by the SPL compiler and execution system. An SPL program consists of the following items:

- . The program source (SPL) file, which is the written sequence or code.
- . The Program Logic Block (PLB) file, which is the compiled source code loaded to the controller.
- . The LIST (LST) file for troubleshooting loaded programs that need to be debugged due to an abort.

These components are explained in the following sections of this chapter.

## 6.5 PROGRAM NAMES

SPL programs must have an associated name that is limited only by the operating system. The valid characters for program names are shown below:

- . A-Z (uppercase letters "A" through "Z")
- . a-z (lowercase letters "a" through "z")
- . 0-9 (numbers "0" through "9")
- . \_ (under bar)
- . (space)
- . . (period)
- . \$ (dollar sign)

Program names are case-insensitive, meaning lowercase letters are treated the same as uppercase letters in program object names (e.g., "abc" is the same as "ABC").

## 6.6 THE .SPL, .PLB AND .LST FILES

SPL programs must reference a Program Logic Block (PLB). The PLB is a binary data file that contains compiled binary pseudocode in a form which can be executed by the controller. This file is created by the SPL Compiler after you create, edit and compile an ASCII text file (i.e., an SPL source file) which contains program logic statements that are easily edited and understood by programmers.

ASCII SPL source code files have the .spl extension and Binary Program Logic Block files (PLBs) have the .plb extension.

The name given to the SPL file can be as long as allowed by the operating system on your computer. However, the controllers impose a limitation on file length. When a PLB is loaded into the GPC's file region, the file name is shortened to 8 characters. If the name was originally longer, only the first 8 characters will be used for the program name.

The source file contains SPL program logic statements which are discussed in detail later in this section. SPL source code can be created and/or edited by using the SPL editor built into SoloPro. The number of lines in an SPL source file is unlimited. Compiled PLBs cannot exceed 8kb in size.

### NOTE

If any errors are generated during the compiling process, a PLB is not created.

You may choose to have the SPL Compilers optionally create a list file during the compile process. The list file is an ASCII text file that contains the source code statements along with the pseudocode and their respective *relative* locations in memory and any error messages generated by the compiler. List files are useful in debugging the execution of SPL programs.

## 6.7 ATTRIBUTES AND REGISTERS

Each SPL program may have up to 255 user-defined properties for storing local data for control sequences. These properties, created as non-standard properties, can be addressed at an operator workstation or web-server.

All programs have 16 registers (**%A-%P**) and a number of program control attributes. To an operator, program registers always begin with a percent sign (%) and program control attributes always begin with a dollar sign (\$). Registers can also be used to hold or present data to an end-user.

## 6.8 COMPILER CONTROL STATEMENTS

Compiler control statements are non-executable directives to the SPL compiler that it uses to control the generation and format of SPL compiler listings and PLBs. The various SPL compiler control statements are summarized below:

```
#TITLE "titletext"
#NOLABELS
#LABELS
#GPC
#PLB08K
#ONESEC
#ENDONESEC
```

Compiler control statements always begin with the pound sign character (#). They must begin in the left-most column.

**#TITLE "titletext"**

The title directive is used to put the specified *titletext* at the top of each page of a compiler list file in order to help identify the program logic. The text string *titletext* must be enclosed in double quotation marks (") and can be up to 79 characters long.

**#NOLABELS**

**#LABELS**

The no labels directive commands the SPL compiler to *not* generate pseudo-code for statement labels in the PLB file. Using this command results in smaller, faster executing PLBs, but eliminates the ability to visually locate labels in the PLB files during troubleshooting. The **#LABELS** command turns on the inclusion of label pcodes in the PLB. In order to conserve RAM and optimize execution, **#NOLABELS** is the default for **#SOLODX**, **#SOLOGX** and **#GPC**. **#LABELS** remains the default for **#SAGE**. The inclusion of label pcodes can be turned on/off throughout the SPL source file.

**#GPC**

The **#GPC** commands identify the target platform for the resulting Program Logic Block (PLB). A summary of statements, terms, operators and features supported by the compiler for each target is shown in Table 1-1. The **#GPC** command can appear any place in the SPL source file, but it is recommended that they appear early in the source file, i.e., directly after the **#NOLIST** and/or **#PLBxx** commands if they are included.

When it executes, the SPL compiler requires a block of RAM in addition to what the executable SPL module uses. The additional block is used to build the PLB. The amount required depends on the maximum anticipated size of the PLB. The **#PLB08K** command set the maximum size of the PLB generated by the SPL compiler at 8192-x. Using **#PLB08K** or **#PLB16K** when it is known that the PLB is less than 8k bytes allocates smaller amounts of RAM from the free-space. If present, the **#PLBxx** statement must be on the second line of the source file if there is a **#NOLIST** command, otherwise it must be on the very first line. There can be no comment lines prior to the **#NOLIST** and/or **#PLBxx** statements.

**#ONESEC**  
**#ENDONESEC**

The **#ONESEC** and **#ENDONESEC** statements are valid only for **#GPC**. They are used to define the extent of a once-a-second routine for use during program execution. There can be at most a single pair of **#ONESEC/#ENDONESEC** statements per SPL source. If the **#ONESEC** statement is present and the **#ENDONESEC** statement is not, then the once-a-second routine is assumed to extend to the end of the PLB. There is an additional set of eight general purpose 32-bit registers (A-P) available for use inside the once-a-second routine. The **#ONESEC** command causes the SPL compiler to make an entry in the PLB header identifying the starting offset of the once-a-second routine for use by the SPL program executor. It also causes the compiler to add a STOP pcode directly preceding the **#ONESEC** statement. The **#ENDONESEC** statement causes the SPL compiler to automatically insert a RETURN pcode directly preceding it.

## 6.9 COMMENTS

All lines in SPL programs that have a semicolon (;) in the left most column are comments. They are for documentation purposes only and are ignored at compile time. The generous use of comment statements within your programs will make them more readable and easier to troubleshoot, especially if *you* are not the person doing the troubleshooting.

As a guideline, the top lines of programs should be reserved for program identification comments. This area may contain information such as:

- . the name of the program
- . the date the program was written
- . the name of the author
- . what the program does
- . the meaning/use of program properties
- . the meaning/use of program registers
- . any assumptions made by the author
- . any input variables used by the program
- . any output values calculated by the program
- . an *edit trail* indicating any modifications made to the program logic, when they were made, and by whom
- . in general, any information that may prove useful to someone looking at the program for the first time

In addition to using comments at the beginning of your program logic, it is also helpful to create a series of comment lines prior to any program segments with logic that may need special explanation. The extra effort you take in adding useful comments to your programs is well worth the benefits you (or someone else) will reap in the future.

## 6.10 LABELS

SPL programs are composed of one or more *statements* which define the actions and logical operations that the program is to take when it is executed. SPL program statements are grouped into *lines* which contain a single program statement. These lines may be labeled with symbolic names to identify them. Typically this is done so that **GOTO** and other branching statements can refer to them.

Labels cannot use any of the reserved names that identify SPL statements. Labels are case-insensitive, meaning that the label **ABC** is treated the same as the label **ABc** or **abc**. Labels must begin in the left most column of the line. Labels may contain up to eight characters or up to eight characters and numbers. Labels can consist of the following:

- . **A** through **Z**
- . **a** through **z**
- . **0** through **9** (not as the first character)
- . **\$**, **.** and **\_** characters.

Labels **cannot** begin with the numbers **0-9**. If a line contains any statement following the label, then the statement must be separated from the label by one or more **TABs** or spaces. Labels may optionally end with a colon character (:), which is not counted in the length of the label.

The lines of an SPL program may be labeled with a symbolic name to identify that line, typically so that **GOTO** and other branching statements can refer to it. Labels may be symbolic and numeric, or symbolic and can be up to 8 characters long. Symbolic labels may not use any of the reserved names that identify SPL statements.

The following program fragment demonstrates several labels:

```

                                C=[A11.PRESENT_VALUE]
LABEL1                            A=[AV1.PRESENT_VALUE]
                                B=A+3.0
                                IF B >32.0 THEN LABEL003
                                C=5
LABEL2:                          IF [A113.PRESENT_VALUE] >72.0 THEN LABEL3
                                SWAIT 30
                                GOTO LABEL2

```

## 6.11 EXPRESSIONS

Expressions are symbolic formulas which represent a chain of arithmetic calculations on data from various sources. Expressions are used to convey values for parameters in many of the primitive statements in the SPL language. SPL expressions can contain an arbitrary number of *terms* and *operators* which may represent mixed mode arithmetic (i.e., integer, floating point and fixed point). SPL automatically performs type coercion on mixed mode values.

Expression evaluation is performed from left to right. Up to six levels of nesting (i.e., the use of parentheses) may be used in expressions to define an order or *precedence* for evaluation. The expression within the innermost set of parentheses is evaluated first from left to right. This procedure continues outward until the expression within the outermost set of parentheses is evaluated from left to right.

Expressions may contain constants, variables, registers, named object attributes, references, tables, built-in functions and arithmetic and logical operators.

In a very general sense, expressions are composed of terms and operators. In the simplest case, an expression is simply a term with no operators. An expression is defined as follows:

***expression ::= term*** or  
***expression ::= term operator expression***

An expression may also be nested within parentheses and used as a term anywhere within an expression. Up to six levels of parentheses may be used.

The syntax for a nested expression is shown below.

**(expression)**

The evaluation of nested expressions occurs first from the innermost set of parentheses and continues outward. Expressions that are at similar levels of nesting are simply evaluated from left to right. The code below shows some complex SPL programming examples using nested expressions.

```
A=MAX(MEAN(B,C,D), MEAN(E,F,G))
H=SQRT((B**2)+(C**2))
[.AA]=[.BB]/255)*[A11.PRESENT_VALUE]+C
```

Because expressions may contain terms that are objects on networks, an expression does not necessarily have to be completely resolved before execution is passed to another program. Such network accessing is processed asynchronously, causing only the program using the value to be delayed until the value has been fetched. This provides for a fair method in dealing with network-intensive programs, and not penalizing other programs by waiting for a network device object value.

Terms in expressions may be one of several possible types, indicating one of several possible sources of a value to be used during expression evaluation. In general, each term has a data type as well as a value. Data types identify the way in which the values are represented numerically, and may imply additional hidden operations or coercions to be performed when arithmetic operations are required between dissimilar types. The types of terms that can be used in expressions are as follows:

- . constants
- . named terms
- . registers
- . program control attributes
- . user-defined program attributes
- . named object attributes
- . references
- . virtual attributes
- . tables
- . functions
- . nested expressions

Each type of expression term is explained in detail in the following sections.

## 6.12 PROGRAM STATEMENTS OVERVIEW

This section is intended to familiarize you with all of the SPL programming statements by organizing them into logical groups based on the functions that they perform. Program statements fall into the following categories:

- . attribute definitions and references
- . assignment statements
- . iteration control, program branching and subroutines
- . program delays
- . printing, logging and alarming
- . job execution
- . spooling
- . trending control
- . program execution control
- . execution error control
- . debugging statements

*Attribute definitions and references* are used to declare user-defined program attributes and save the values of attributes to the program's attribute initial value (INI) file.

*Assignment statements* are used to assign the value of an expression to a variable. This type of program statement is characterized by the use of an equal sign (=).

*Iteration control, program branching and subroutines* are statements perform a statement or group of statements some number of times, change the order in which the logic is executed, or transfer program control to another portion of the program (a subroutine).

*Program delays* are statements which suspend program execution, either for a set amount of time or until certain conditions are met.

*Printing, logging and alarming* refers to statements that give you the ability to print information to a port, log information to a file, or generate formatted alarms of definable alarm classes.

*Job execution* refers to statements that give you the ability to execute any SAGE<sup>MAX</sup> job from within the SPL program execution environment.

*Spooling* refers to commands which offer the ability to send specified files to the printer.

*Trending control* refers to program statements that can control the execution of trends from a program.

*Program execution control* refers to statements that can start, stop and prepare programs to be executed.

*Execution error control* refers to program statements that allow you to define a course of action for PEX when network access errors occur.

*Debugging statements* refer to programming statements that can be used to aid in the diagnosis of program logic errors.

Each SPL programming statement is individually explained, including sample SPL statements in the following pages.

Table 6-2 Program Statements

Format	Description
<i>variable = expression</i>	assignment statement
ATTR <i>definition, datatype</i>	declares a new attribute and PUP datatype.
ERRORABORT	trap condition - abort on errors
ERRORWAIT	trap condition - wait until no error
<i>symbol EQU expression</i>	symbolic equate statement
GOSUB <i>label</i>	go to internal subroutine
GOTO <i>label</i>	unconditional branch
IF <i>expr</i> THEN <i>label</i>	conditional branch if <i>expr</i> is true
IF <i>expr</i> THEN <i>label1</i> ELSE <i>label2</i>	conditional branches if <i>expr</i> is true or false
LOOP <i>register,label</i>	iteration control
MWAIT <i>expression</i>	wait a certain amount of minutes
ON <i>expression</i> GOTO <i>label0,label1...labeln</i>	indexed conditional branches
ONERROR <i>label</i>	trap condition - branch if error occurs
RETURN	return from a subroutine
SECTION <i>number</i>	section marker used for debugging
STOP	halt execution of this program
SWAIT <i>expression</i>	wait a certain amount of seconds
WAIT ( <i>expression</i> )	wait until an expression is true, then go on

## 6.13 ASSIGNMENT STATEMENTS AND EQUATES

### 6.13.1 STANDARD VALUE ASSIGNMENT

***variable = expr***

SPL allows various forms of value assignment statement. In each case, a variable on the left side of the = (equal sign) is assigned the new value dictated by the expression on the right side. The right side expression produces a value and a data type. Because automatic data type *coercion* may occur during evaluation, the data type of the expression may not match the data type of the variable on the left side. In this case the data type and value from the expression *may* have to be coerced into the variable's data type according to certain rules. The table below summarizes the conversions in general:

Left Side	Right Side	Effect
fixed	float	left=FIX(right)
float	fixed	left=FLOAT(right)
fixed*	fixed	left=RETYPE(right)

\*Different fixed data type.

Integers and time data types are treated as fixed types. The table above does not reflect that there are 20 distinct types of fixed types, i.e. 10 decimal point positions each for signed and unsigned types.

When the left side variable is a local program attribute, coercion of the expression into the proper data type is done automatically by PEX. When the left side variable is a register, unless the data type of the result is converted according to the table above, the data type of the register is automatically changed to the data type of the result. When the left side variable is any other type of object, the result *must* be converted according to the table or incorrect values may be assigned to the variable. For example, if the left side variable is a point whose data type is F9H (xxxxxxx.xxx) and the expression has a data type of F7H (xxxxxxx.xxxx) then a RETYPE (F9H) must be done so that the value assigned to the variable is not 10 times too large (in this case.)

There are several forms of assignment statements that may be used in SPL. These are summarized below:

<i>register = expression</i>	A = B+C
<i>;programattribute = expression</i>	;CV = B+C
<i>namedobject = expression</i>	OAT = B+C
<i>namedobject = expression</i>	[ZONE TEMP] = B+C
<i>namedobject = expression</i>	[1STFLOOR] = B+C
<i>\objecttype\namedobject = expression</i>	\VR\OAT = B+C
<i>namedobject;attribute = expression</i>	[LOOP;SP]= B+C
<i>\objecttype\namedobject;attribute = expression</i>	\PT\LOOP;SP = B+C
<b>REF(expression) = expression</b>	REF(6) = B+C
<b>&amp;tablename(expression) = expression</b>	&CLAIREX(29) = B+C

<code>UNS(x1,x2,x3,x4,x5,attribute) = expression</code>	<code>UNS(1,0,0,0,FB00h,CV) = B+C</code>
---	--

At first there would appear to be a conflict in syntax between local attribute references that are used as variables on the left hand side of assignment statements, e.g., `;AT=expression`, and comments since both begin with semicolons. The difference in syntax between the two is that comments begin in the left most column and local program attribute references used as variables must have at least one leading space or tab. In order to avoid confusion, local program attribute references can be enclosed in brackets, i.e. `[:AT]=expression`.

### 6.13.2 EQU

#### *symbol EQU expression*

**EQU** (Equate) provides a simple method to assign substitute names to commonly used point references in an SPL program, providing the ability to easily read and interpret an SPL program in a more basic form. **EQU** is a symbolic equate in its rawest form.

**EQU** statements must be defined in a program *before* they are used, because the compiler considers all terms that are not SPL keywords, numeric values or SPL symbols to be object names. The symbol part of the **EQU** statement can be up to 16 characters in length, which must all be printable characters (A-Z, 0-9, !, @, etc.) and cannot begin with a digit. The right-hand side of the expression can be up to 32 printable characters and can be a programmatic expression or point data location (e.g. FE01;CV).

The code below illustrates the use of the **EQU** statement in an SPL programming example:

```
FANSTATUS EQU [FE01;CV]
FANOUTPUT EQU [FB02;CV]

;start of program
L0: SWAIT 1
    FANOUTPUT = 1
    SWAIT 5
    FANOUTPUT = 0
```

## 6.14 ITERATION, BRANCHING AND SUBROUTINES

### 6.14.1 GOTO STATEMENT

#### **GOTO label**

where:

*label* is the label of the point to which program execution will be switched

The **GOTO** statement is an unconditional branch statement that causes program logic to jump to some other location that is identified by a label.

The code below illustrates the use of the **GOTO** statement and shows a sample SPL programming example. It may increase the readability of your program logic if you add a blank line after **GOTO** statements.

```

      :
L1:   C = A+B
      GOTO L3

      L2:   C = B-A
L3:   D = C*2
      :
```

### 6.14.2 IF... THEN... {ELSE...} STATEMENT

#### **IF expr THEN label1 {ELSE label2}**

where:

*expr* is the logical expression which determines conditional branching behavior

*label1* is the label to jump to if *expr* evaluates to *true*

*label2* is the label to jump to if *expr* evaluates to *false* (optional)

The **IF... THEN... {ELSE...}** statement is a conditional statement that causes the program logic to jump to some other location identified by a label if a certain condition is true. If the condition is false, execution *falls through* to the next sequential statement. If the optional **ELSE** statement is included, then program execution will jump to the label following the **ELSE** statement if the condition evaluates to *false*.

The code below illustrates the use of the **IF... THEN... ELSE...** statement and shows its usage in an SPL programming example.

```

      IF (DAYOFWEEK==SUN) THEN L3
L0:   IF (A>B) THEN L1 ELSE L2
L1:   C=A+B
      GOTO L3

      L2:   C=B-A
L3:   D=C*2
      :
```

## 6.14.3 ON... GOTO... STATEMENT

**ON** *expr* **GOTO** *label0,label1,label2,label3,....*

where:

*expr* is the expression which determines which label is selected

*label0,label1,label2,label3,....* are the labels of the sections to which program control can be switched

The **ON/GOTO** statement is a conditional statement that identifies a series of indexed labels to which PEX transfers control based on the value of an expression. The code below illustrates the use of the **ON/GOTO** statement.

```

                ATTR ER,07
                :
                ON INT(B-10) GOTO L0,L1,L2
                :ER=1
                PRINT 13,226,"Unsuccessful."
                GOTO DONE

L0:             D = (C+1)/2
                GOTO MERGE

L1:             D = (C+20)/2
                GOTO MERGE

L2:             D = (C+50)/2
MERGE:         PRINT 13,226,"Success. D=%?%",D
DONE:         STOP

```

The indices of the **ON/GOTO** statement are zero-based. In addition, if an index evaluates to a number that is greater than the number of indices, program execution continues with the next line of the program.

## 6.14.4 LOOP STATEMENT

**LOOP** *register,label*

where:

*register* is the number of times the loop is to be executed

*label* is the program label to which execution will jump

The **LOOP** statement is an iteration control statement that performs a “decrement register and jump if not zero” function using a specified register and a program label. The **LOOP** statement is a combination of an assignment statement (e.g., **A = A-1**) and a conditional statement (e.g., **IF A>0 THEN Continue**).

The code below illustrates the proper use of the **LOOP** statement in a sample SPL programming example.

```

A = 100
B = 0

```

```
CALC:      B = REF (A-1)+B
           LOOP A, CALC
           REF (100)=B/100
```

#### 6.14.5 GOSUB STATEMENT

##### **GOSUB** *label*

where:

*label* is the text label which specifies the starting point of the subroutine

The **GOSUB** statement is used to call a subroutine *in the current PLB*. A **RETURN** statement is used to terminate the internal subroutine and return execution control to the statement directly following the **GOSUB**. The subroutine name is actually a label for which all the naming conventions apply.

The code below illustrates the syntax of the **GOSUB** statement and shows its use in a sample SPL program segment:

```
          ATTR AR,0FAH
          A=65
READIT:   D=&DuctDiam(A-1)
          GOSUB AREACALC
          &DuctArea(A-1) = ;AR
          LOOP A, READIT
          :
AREACALC: ;AR = PI*(D*D)/4
          RETURN
```

#### 6.14.6 RETURN STATEMENT

##### **RETURN**

The **RETURN** statement must be used in conjunction with the **CALL** and **GOSUB** statements.

## 6.15 PROGRAM DELAYS

### 6.15.1 SWAIT AND MWAIT STATEMENTS

#### **SWAIT** *expr*

where:

*expr* is the number of seconds to delay program execution

#### **MWAIT** *expr*

where:

*expr* is the number of minutes to delay program execution

The **SWAIT** and **MWAIT** statements are used to cause a timed delay in program execution. These statements each have a single argument which represents a number of seconds or minutes (respectively) that must pass before program execution continues. The time delay can be viewed as it counts down from the **\$D** program control attribute. This attribute shows all time delays in seconds. Once the delay reaches zero, the next program statement is executed.

The code below illustrates the proper use of the **MWAIT** and **SWAIT** statements. Sample SPL programming examples are also shown.

```
L0:      IF (SWITCH==1) Then L1
          MWAIT 5
          GOTO L0

L1:      [PROG1;MN]=[PROG2;SP]+5.0
```

### 6.15.2 WAIT STATEMENT

#### **WAIT** *expr*

where:

*expr* is the logical expression that will determine when the **WAIT** will finish

The **WAIT** statement is a conditional statement that halts further program execution until the expression specified in the argument is true.

The code below illustrates the syntax of the **WAIT** statement and shows a sample SPL programming example.

```
L0:      WAIT $ALARMS
          CALL Notify

L1:      CALL Clear, STICK
          IF $ALARMS THEN L1 ELSE L0
```

## 6.16 EXECUTION ERROR CONTROL

### 6.16.1 ERRORABORT STATEMENT

#### **ERRORABORT**

The **ERRORABORT** statement is an error control statement that causes the program executor to abort the program when any trappable or non-trappable error is detected. (See also *Section 6.16.2:ERRORWAIT Statement* and *Section 6.16.3:ONERROR Statement*).

There can be multiple **ERRORABORT** and **ERRORWAIT** statements within a program. This allows the aborting of errors to be turned on and off. Unless an **ERRORWAIT** statement is included in a program, the **ERRORABORT** statement is in effect.

All errors that are not trappable (e.g., no such object name, invalid operation, etc.) will always cause the program to be aborted.

The code below illustrates the syntax for the **ERRORABORT** statement and shows its use in a simple SPL program segment:

```
ERRORABORT  
[FD01;CV]=55.255
```

### 6.16.2 ERRORWAIT STATEMENT

#### **ERRORWAIT**

The **ERRORWAIT** statement is an error control statement that allows the programmer to specify what PEX should do when it encounters a trappable error. If the **ERRORWAIT** statement is included in a program and PEX detects a trappable error, then the statement that caused the trappable error is re-executed forever until the error condition no longer exists

There can be multiple **ERRORWAIT** and **ERRORABORT** statements within a program. This allows the aborting of errors and error waiting to be staggered throughout the program. Unless an **ERRORWAIT** statement is included in a program, the **ERRORABORT** statement is in effect.

The code below illustrates the syntax of the **ERRORWAIT** statement and shows it being used in an SPL program segment:

```
ERRORWAIT  
[FE01;CV]=55.255  
:
```

### 6.16.3 ONERROR STATEMENT

#### **ONERROR label**

where:

*label* is the label of the code to be executed when a trappable error occurs

The **ONERROR** statement identifies a label to which PEX transfers control whenever it detects a *trappable* error (see **Appendix B**). The **ONERROR** statement is in effect only for the statement that precedes it. (See also *Section 6.16.1:ERRORABORT Statement* and *Section 6.16.2:ERRORWAIT Statement*). When an error is detected, the error code is placed in the program's **\$E** control attribute by PEX. **ONERROR** statements take precedence over **ERRORWAIT** statements. The **\$E** program control attribute should be reset to zero before leaving the error code handler.

The code below illustrates the syntax of the **ONERROR** statement and shows an SPL programming example.

```
Getit:      ;$E = 0
            A = ZONE_TEMP;CV
            ONERROR Err
L1:         B=A+10
            :
Err:        IF (;$E<>5) THEN END
            A=72.0
            GOTO L1
End:       STOP
```

#### NOTE

The **ONERROR** statement can only be used with trappable errors such as a timeout, a CRC or checksum error, NAK responses, data rejection, temporarily blocked states, dialer busy states and failed to connect errors. Any other program execution errors cause the program to abort.

## 6.17 DEBUGGING STATEMENTS

### 6.17.1 SECTION STATEMENT

#### **SECTION** *number*

where:

*number* is the number designation given to the section

The **SECTION** statement is a debugging statement that stores the *number* argument in the **\$\$** program control attribute of the program. This command can be placed strategically at multiple locations in the program to be debugged. By using unique *numbers* in the statements, you can track the progress of the program through various logical sections by monitoring the **\$\$** program control attribute.

The code below illustrates the syntax of the **SECTION** statement and shows an SPL programming example:

```
SECTION 1
A = REF (0)
L1: SECTION 2
    B = B + REF (A-1)
    LOOP L1
SECTION 3
    :
```

## 6.18 PROGRAM CONTROL ATTRIBUTES

All program control attributes begin with the '\$' and are referred to as the 'dollar attributes'. The program control attributes are listed below, along with a brief description. Depending on the device you are using, the existence of specific program control attributes differs. Please reference device user documentation for additional information.

Table 6-3 Program Control Attributes

Control Attribute	Meaning
\$	Program status 0=Stop 1=Run 2=Unloaded 3=Abort 4=Time Delay 5=Restart 6=Load Request 7=Unload Request 8=Abort Request 9=Wait for fetch 10=Reload Request
\$\$	program execution is stopped program is executing program logic is not loaded into RAM program has aborted due to a run-time error program is waiting for an WMAIT or SWAIT statement to timeout initializes the program and starts executing it from the beginning request to load the program into RAM and begin execution request to stop the program and unload it from RAM PEX has encountered a run-time error and is aborting the program program is waiting for the completion of a network access PEX has received a request to unload, reload and start a program
\$D	Number of seconds remaining in time delay
\$E	Error code (0=no error) (Refer to Appendix B for a complete list of error codes)
\$S	Current section number
\$C	Program location counter in hexadecimal of next statement to be executed
\$W	Wait/Abort on trappable errors such as timeouts and CRC errors (see Appendix B)

The \$\$ attribute indicates the program's current operating status. The \$\$ attribute can have one of the following values:

Value	State	Description
0	Stop	program execution is stopped
1	Run	program is executing
2	Unloaded	program logic is not loaded
3	Abort	program has aborted due to a run-time error
4	Time Delay	the program is waiting for an MWAIT or SWAIT to timeout
5	Restart	re-initializes the PCB and starts execution at beginning of program

Value	State	Description
6	Load Request	requests that the program logic be loaded and the program started
7	Unload Request	requests that the program be stopped and the logic be unloaded
8	Abort Request	PEX has encountered a run-time error and is aborting the program
9	Wait for fetch	the program is waiting for the completion network object read/write

*Stop* indicates that the program has stopped and is no longer executing its program code. *Run* indicates that the program is in the process of executing its program code. *Unloaded* indicates that the program logic has not yet been loaded into memory. *Abort* indicates that the program has stopped due to a run-time error. *Wait for time* indicates that the program has encountered an **MWAIT** or **SWAIT** statement in its logic and is in a wait state. *Restart* indicates that the program has been initialized and is going to start executing from its beginning. *Load request* indicates a signal for the program to be loaded into memory (i.e., RAM) and to begin execution. Conversely, *unload request* indicates a signal for the program to stop execution and unload (remove) itself from RAM. *Abort request* indicates the state prior to abort when the program executor (PEX) encounters a run-time error and signals that it should be aborted. *Network access* indicates that the program had executed either a network read or network write request and is currently in the process of waiting for the network transaction to be completed. *Reload request* is used to unload a program, then reload and start it.

Not all transitions from one **\$\$** state to another are valid. The following matrix summarizes the valid state transitions for the **\$\$** attribute.

The **\$E** control attribute specifies the error code number of the previously executed program statement. Normally this attribute equals zero (no error).

This special control attribute can be read as an attribute from the program and can be used to determine a course of action should an error occur (i.e., **\$E** <> 0). The example below illustrates the use of the **\$E** control attribute in a sample SPL program segment.

```

GETIT:      ;$E = 0
           A = [ZONE_TEMP;CV]
           ONERROR ERR

ERR:       IF ;$E==5 THEN GETIT
           STOP

```

The **\$\$** control attribute is another special control attribute which reflects the current section number of the program. The **SECTION** statement is used to set the **\$\$** attribute to any integer value (refer to *Section 6.17.1:SECTION Statement*). This control attribute can be used in diagnosing errors in your program logic. By using **SECTION** statements at strategic locations in the logic (e.g., before and after loops, conditional statements, calls to subroutines, etc.), you can check the progress of the program execution. By examining the **\$\$** control attribute through OPI monitor/modify, you can determine if a particular segment of code is getting executed. The example below illustrates the use of **SECTION** statements so that the **\$\$** control attribute can be examined.

```
SECTION 1
```

```
CALL INITIALIZE
SECTION 2
CALL CALC_LOOP
SECTION 3
CALL PROCESS_LOOP
SECTION 4
CALL SUBMIT_JOB
SECTION 5
STOP
```

For logic errors that are especially difficult to diagnose, you may choose to use the single step mode of execution (control attribute **\$1**) in conjunction with the **\$E** and **\$S** control attributes. When set to single step mode (**\$1=1**), the program is executed one line at a time. The program must be set to the **RUN** state (**\$\$=1**) after each line of the program is executed. In some complex programs, this may be a helpful way to determine if your program logic is doing what you really want it to do or if you are encountering a program error. Using single step mode may also make it easier to follow the logic of large programs at a statement-by-statement level.

The **\$\$** control attribute reflects the current state of the program. This attribute can assume one of eleven values representing one of eleven possible states for the program. These states are:

- . 0 - stop
- . 1 - run
- . 2 - unloaded
- . 3 - abort
- . 4 - wait for time
- . 5 - restart
- . 6 - load request
- . 7 - unload request
- . 8 - abort request
- . 9 - wait for fetch
- . 10 - reload request

*Stop* indicates that the program has stopped and is no longer executing its program code. *Run* indicates that the program is in the process of executing its program code. *Unloaded* indicates that the program logic has not yet been loaded into memory. *Abort* indicates that the program has stopped due to a run-time error. *Wait for time* indicates that the program has encountered an **MWAIT** or **SWAIT** statement in its logic and is in a wait state. *Restart* indicates that the program has been initialized and is going to start executing from its beginning. *Load request* indicates a signal for the program to be loaded into memory (i.e., RAM) and to begin execution. Conversely, *unload request* indicates a signal for the program to stop execution and unload (remove) itself from RAM. *Abort request* indicates the state prior to abort when the program executor (PEX) encounters a run-time error and signals that it should be aborted. *Network access* indicates that the program had executed either a network read or network write request and is currently in the process of waiting for the network transaction to be completed. *Reload request* is used to unload a program, then reload and start it.

The **\$D** control attribute reflects the number of seconds remaining in a time delay imposed by either an **SWAIT** or **MWAIT** statement. When **\$D<>0**, **\$\$=4** (wait for time).

The **\$C** control attribute indicates the program location counter. As the program executes, **\$C** changes, reflecting the relative memory location of the next program statement to be executed. **\$C** is used primarily for low-level troubleshooting and diagnosis of program errors.

The **\$1** control attribute is used to set the single step mode of execution of a program. When set to 0, program execution continues normally. If this attribute is set to 1 (single step mode), program execution stops after each program line is executed. Once program execution is stopped, you can examine the other control attributes, registers and user-defined program attributes. This may be very helpful in troubleshooting and diagnosing hard-to-find errors in your program logic. The next line of program code can be executed by setting **\$\$=1** (the RUN state).

The **\$W** attribute is used by PEX to control the execution of the program. Although their meanings are summarized in Table 6-3, the programmer and/or operator normally does not need to reference them.

## 6.19 WORKING WITH CHANNEL ATTRIBUTES

Accessing objects and properties in PUP using SPL can be done in a variety of methods. The following section reviews the various methods of how to access objects and properties.

### 6.19.1 ADDRESSING CHANNEL ATTRIBUTES

When addressing object properties in SPL, the following format must be used:

**channel;attribute** where

- . channel references the hex channel assignment.
- . attribute references the two-character attribute.

A semicolon ( ; ) must separate the channel and attribute references.

The following examples are illustrated:

```
FE01;CV
;references Universal Input 1's Current Value
FB01;CV
;references Binary Output 1's Current Value
U1234_FE01;CV
;references Universal Input 1's Current Value on Unit 1234
```

### 6.19.2 WRITING TO TEXT-BASED ATTRIBUTE VALUES

As of SBC-GPC firmware revision v2.06, SPL-based writes to text-based values (like IC and IA) are now supported via a wider range of non-text values. Examples are provided below on how to achieve this capability

Attribute inputs that require text writes of values such as "CV" can now occur as:

```
F321;A1=0x4356
;This is "CV" in hex
;
F321;A1=17238
;This is "CV" in hex
```

Channel inputs that require text writes of hex values such as "FE01" can now occur as:

```
F321;I1=0xFE01
;This is the actual hex channel value
;
F321;I1=65025
;This is the actual hex channel value converted to decimal
;
F321;I1=0x46453031
;This is "FE01" in hex
;
F321;I1=1178939441
;This is "FE01" in hex converted to decimal
;
```

### 6.19.3 ADDRESSING USER-DEFINED ATTRIBUTES

To reference user-defined attributes created at the top of your program, the following format must be used:

**[;attribute]**

where

- . ;attribute is the custom attribute declared.

By referencing no object, SPL will look inside its own program for the attribute reference.

```
#GPC
;
      ATTR AA,254
      PROP BB,254
;
L0:   ;AA=13
      ;BB= 16
```

### 6.19.4 PEER-TO-PEER ADDRESSING

SPL allows users to perform peer-to-peer transactions on the PUP network that the controller resides on.

To address a channel attribute from a remote device, the following format must be used:

**[####\_channel;attribute]**

where

- . **####** is the Unit ID of the device you wish to access.
- . **channel** is the hex channel assignment of the object you are attempting to access.
- . **attribute** is the two character attribute assignment..

The following example illustrates this function:

```
#GPC
;
L0:   A = U1234_FE01;CV
```

When accessing information from remote devices, users should place SWAIT statements of about 3 seconds between each peer-to-peer network transaction that is made. This allows for the device to receive the token from the network. If you declare ERRORWAIT, SPL will trap a PUP communication timeout if encountered.

Please note that GPC family devices can only access PUP information on the local network it is connected to.

---

# SECTION 7: CONTROL LOOPS

*This section provides information regarding control loop objects that reside in the GPC platform, including Analog PID Control Loops, Floating Point Control Loops, and Thermostatic Control Loops.*

## IN THIS SECTION

Control Loops Overview.....	7-3
Programming Concepts and Techniques .....	7-3
Analog Output Control Loops.....	7-4
Basic Setup.....	7-4
Proportional Control Setup.....	7-5
Deadband Configuration .....	7-6
Reset Control Setup.....	7-8
Soft Start Setup .....	7-11
Enabling the Control Loop .....	7-11
Floating Point Control .....	7-13
Basic Setup.....	7-13
Proportional Control Setup.....	7-14
Deadband Configuration .....	7-15
Reset Control Setup.....	7-17
Calibration .....	7-20
Interlocking and Fire Positioning.....	7-22
Enabling the Control Loop .....	7-23
Thermostatic Control.....	7-24
Basic Setup.....	7-24
Enabling the Control Loop .....	7-25



## 7.1 CONTROL LOOPS OVERVIEW

The GPC platform provides built-in loop objects which can be setup to provide control to outputs of the GPC. Control Loops provided by the GPC can be programmed to directly affect the status of analog or binary output.

Three types of control loop channels are provided within the GPC, including:

1. Analog PID - uses standard PID control to provide an analog signal value.
2. Floating Point - uses standard PID control to provide on/off floating point control signals.
3. Thermostatic - uses enhanced boolean logic (TSTAT logic) to provide an on/off control signal.

An explanation of each control loop channel type, along with notes on setup and configuration are provided in the sections below.

### 7.1.1 PROGRAMMING CONCEPTS AND TECHNIQUES

To enhance your programming experience, the following are a few helpful concepts and techniques to keep in mind when using these objects.

#### 7.1.1.1 MAKE THE (ON) CHANNEL NAME UNIQUE

The GPC supports the ability to allow each channels name to be assigned a custom value. By default, the software uses generic names for channels. For ease of programming and flow, it is strongly recommended that you change the **(ON) Channel Name** of any used control loop objects. This allows you not only to keep better track of which objects have been used, but also allows you to easily troubleshoot your linked logic.

## 7.2 ANALOG OUTPUT CONTROL LOOPS

Proportional + Integral + Derivative (PID) represents a method of control that controls equipment according to a set point in proportion to the value of a measured variable. It accounts for the amount of error (difference between the measured variable and the set point) and the continued presence of error.

### 7.2.1 BASIC SETUP

To initially use a control loop, you must first setup and configure basic properties of the control loop.

#### 7.2.1.1 CONTROL ACTION AND OUTPUT LIMITS

The **(SG) Control Action** attribute specifies the control action for the control loop. When **(SG) Control Action** = 0 (normal), a positive error causes an increase in output. When **(SG) Control Action** = 1 (reverse), a positive error causes a decrease in output. This point determines the response of the loop output to the kind of error. If the output action is to be increased (toward max) when the error is positive, set **(SG) Control Action** to normal (0). If the output action is to be decreased (toward min) for positive error, set **(SG) Control Action** to reverse (1). **(SG) Control Action** is also used during schedule control)

The minimum and maximum limits of the control loop may also be configured if desired. The **(OL) Minimum Output Limit** and **(OH) Maximum Output Limit** attributes define the minimum and maximum limits of the output range for the control loop. The **(PO) Percent Output Value** will be scaled between the limits you have defined.

#### 7.2.1.2 MEASURED VARIABLE CONFIGURATION

The **(IC) Input Channel** and **(IA) Input Attribute** attributes specify the channel and attribute to be used as the loop measured variable. It specifies the input to be used for the control loop's measured variable. Any data point within the GPC can be used as the measured variable by configuring these attributes appropriately.

#### 7.2.1.3 SETPOINT CONFIGURATION

Each control loop contains a **(SP) Loop Setpoint** attribute, defining the control setpoint that is used for a specific schedule mode. The setpoint is expressed in the same kind of measurement units (engineering units) that the measured variable uses (e.g., degrees, cfm, inches of WC, etc.).

The **(SU) Unoccupied Setup/Setback** attribute allows the calculated setpoint to be setback during unoccupied periods.

When the control loop is later enabled (using **(CE) Enable Control Loop**), the **(CS) Calculated Control Setpoint** will reflect which setpoint is currently active.

The **(P0) Permanent Stat Override Enabled** attribute allows the user to select whether a setpoint override made from a STAT should be permanent. When enabled, the override is made permanent and persists through controller reboots. Corresponding with the addition of this new variable, the **(TS) Thermostat SP Adjustment** will only effect the setpoint when the controller is in an occupied or warmup schedule mode.

#### 7.2.1.4 SCHEDULES TO FOLLOW

The **(SM) Schedules to Follow** attribute allows control loop to reference a schedule and transition through programmed set points appropriately. Each bit in **(SM) Schedules to Follow** corresponds to any of the schedule channels within the GPC, as well as STAT Overrides, and other options. These bits are summarized in Table 7-1.

Table 7-1 : Schedules to Follow

SM bit	Schedule
0	F901
1	F902
2	F902
3	F903 4
4	F904
5	F905
6	F906
7	F907
8	F908
9 - 21	STAT Override Ulx
22	Host Schedule (F900;HO)
23	F900;CV
24	Occupancy Extension (FC01)

If you do not want to use schedule control, set all of the bits in **(SM) Schedules to Follow** to 0.

### 7.2.2 PROPORTIONAL CONTROL SETUP

The **(PB) Proportional Control Band** specifies the input variable range over which the output value is proportional to the error value (i.e., changes in the measured variable result in proportional changes in the output signal). The proportional band is centered around setpoint for the loop. This point is expressed in the same kind of measurement units (engineering units) that the measured variable uses. For example: degrees, cfm, inches of WC.

To determine **(PB) Proportional Control Band**, first decide how closely the GPC must control to the setpoint. For instance, if the setpoint is 72°F, then an acceptable control range might be within two degrees of the setpoint. This control range can be expressed as a band centered on the setpoint: from 70° to 74°, or 4 degrees *proportional band*. Refer to Figure 7-1 and Figure 7-2.

For normal acting control loops (see Figure 7-1), the **(PO) Percent Output** attribute is set to maximum output when the input variable equals the setpoint plus half of the proportional band (**CS + PB/2**). The percent output is set to minimum output when the input variable equals the setpoint minus half of the proportional band (**(CS) Calculated Control Setpoint - (PB) Proportional Control Band / 2**). These associations are reversed for reverse acting control loops. **PO** will be midway between minimum and maximum output when the measured variable is equal to the control setpoint (**CS) Calculated Control Setpoint**. The opposite would be true for reverse acting control loop as shown in Figure 7-2.

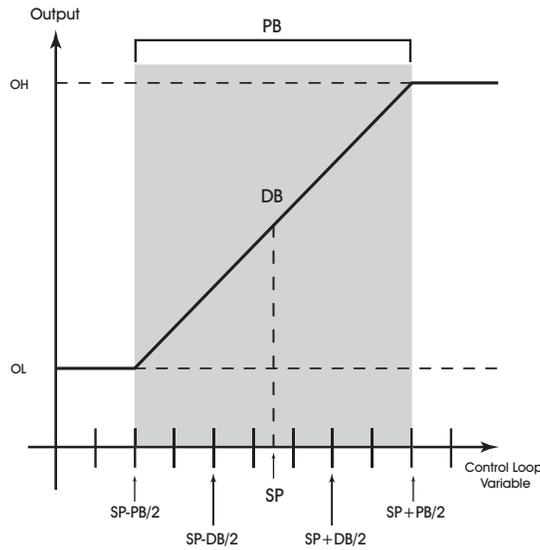


Figure 7-1: Proportional Band for Normal Acting Control ( $SG = 0$ )

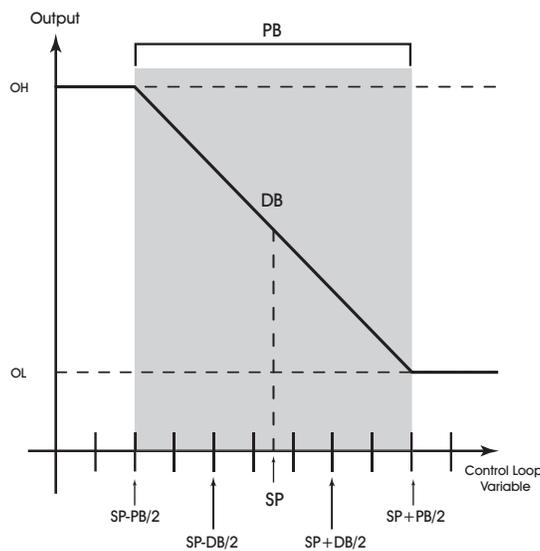


Figure 7-2: Proportional Band for Reverse Acting Control ( $SG = 1$ )

Proportional only control produces cycling, and its performance changes when the measured environment changes. The way to eliminate cycling and to compensate for load changes is to use *integral* action, the “I” part for PID control.

### 7.2.3 DEADBAND CONFIGURATION

The **(DB) Desired Control Deadband** attribute specifies the deadband within the proportional control band in which the output remains constant at a point midway between maximum output and minimum output. By specifying a deadband that is greater than or equal to the resolution of the loop measured variable, you eliminate the possibility of cycling around the setpoint. The value of the deadband should

never exceed the proportional band. If the deadband is greater than the proportional band, then the control loop will not have proportional control.

The deadband is used to specify an input variable range within the proportional band. The size of the deadband should be based on the loop measured variable. When the value of the measured variable is within this dead band, the output signal remains constant at the midpoint of the minimum/maximum range.

The point that deadband is centered on one of the four defined set points to create the actual control dead band. When the value of the loop measured variable is within  $\pm(\text{DB})$  **Desired Control Deadband/2** of the setpoint, the GPC assumes that it has reached the setpoint. Refer to Figure 7-3.

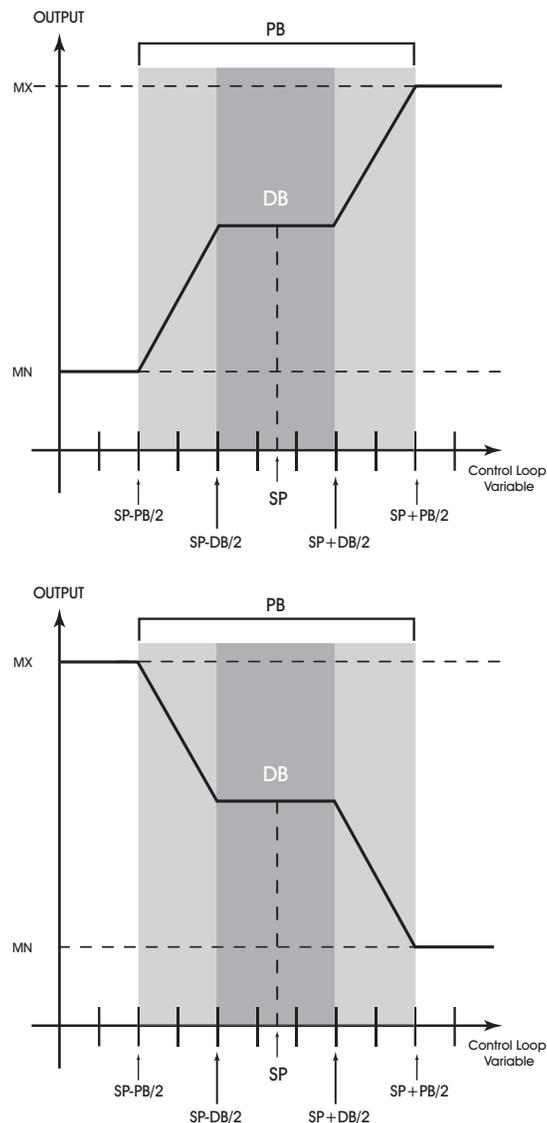


Figure 7-3: Normal Acting (above) and Reverse Acting (below), Proportional Control Output Response Showing a Dead Band Centered Around the Setpoint (SP)

By entering a value in deadband that is greater than the resolution of the measured variable sensor, you create a deadband that allows the GPC to effectively reach setpoint. Be sure that the deadband selected does not exceed the size of the proportional band.

CAUTION	
	<p>Never change <b>(DB) Desired Control Deadband</b> to a value greater than half of the proportional band <b>(PB) Proportional Control Band</b>. Doing so will eliminate the effects of PID control, resulting in on/off control.</p>

#### 7.2.4 RESET CONTROL SETUP

The **(MR) Maximum Amount to Reset Setpoint** attribute specifies the maximum amount to reset the loop setpoint (**SP**) based on when reset is being used. **(CS) Calculated Control Setpoint** takes into account the use of the maximum reset specified in **(MR) Maximum Amount to Reset Setpoint**.

The **(RC) Reset Channel** and **(RA) Reset Attribute** specify the channel and attribute to be used as the Reset Variable.

The **(RS) Reset Setpoint** attribute specifies the value at which the reset action begins. When the value of the reset variable exceeds **(RS) Reset Setpoint**, reset action will be used in determining the calculated setpoint. Just as **SP** is the proportional control setpoint for the measured variable specified in **(IC) Input Channel** and **(IA) Input Attribute**, **(RS) Reset Setpoint** is the reset control setpoint for the value of the reset variable selected by **(RC) Reset Channel** and **(RA) Reset Attribute**.

The **(RL) Reset Limit** attribute specifies the value at which maximum reset is used. When the value of the reset variable is equal to **RL**, the **(MR) Maximum Amount to Reset Setpoint** is used in determining the calculated setpoint (reported through attribute **(CS) Calculated Control Setpoint**).

The relationship between **(RL) Reset Limit** and **(RS) Reset Setpoint**, as well as the sign (+ or -) of **(MR) Maximum Amount to Reset Setpoint**, determines how changes in the reset variable specified by **(RC) Reset Channel** and **(RA) Reset Attribute** affect the calculated control setpoint **(CS) Calculated Control Setpoint**. Refer to Figure 7-4.

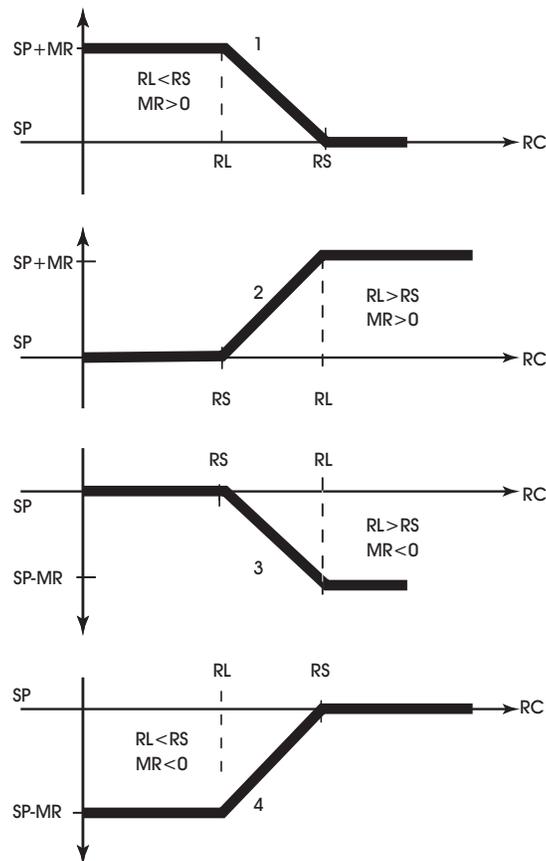


Figure 7-4: Four Forms of Reset Action

With appropriate values entered for these properties, the GPC will provide simple closed loop feedback proportional control. This means that the actual measured performance of the control (from the measured variable input) is fed back to the controller and is compared with the effective setpoint for the loop. Any difference between the actual value of the measured variable and effective setpoint values is called error.

One problem with proportional only control is the changes in loop performance that occur when the condition being measured by the input sensor changes (e.g., the measured temperature changes when a door is opened and the room or space is flooded with cold air). As the loop environment changes, the proportional only control loop begins to cycle around an offset from the setpoint. Figure 7-5 illustrates the performance of a typical loop under proportional only control.

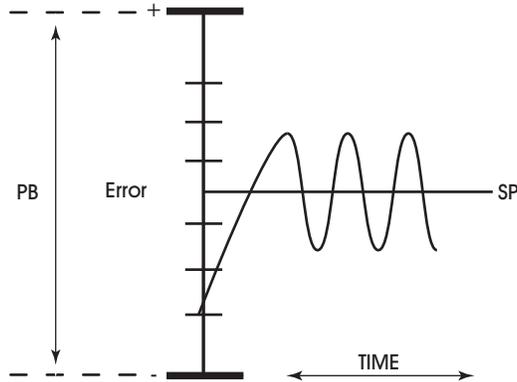


Figure 7-5: Proportional Only Control

Rather than responding exclusively to the loop error from moment to moment as is the case with proportional action, integral action is based on a summation of the error that has occurred over some period. This error sum is used to reset, or modify, the response of the control loop (output) based on a running average of the error. The amount of time over which the error averaging is accumulated is called the *reset period*.

The **(RP) Reset Period** attribute specifies the reset period (in seconds) over which the error history is accumulated. If **(RP) Reset Period** = 10 seconds with a constant error of 2.0, then the error history would increase by 0.2 every second. In five seconds, the error history would be 1.0. At the end of ten seconds, the error history would be 2.0. Setting **(RP) Reset Period** to 0 disables integral action making the loop proportional only. The longer **(RP) Reset Period** is, the less effect it has on the control response. Figure 7-6 shows the response of a typical control loop when integral action is used in addition to proportional action (PI control). A value of 0 disables the reset period.

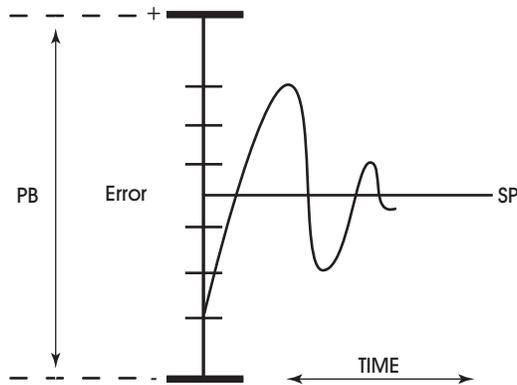


Figure 7-6: Proportional + Integral (PI) Control

At the start-up of the loop or following a change in setpoint (see Figure 7-6), the error is fairly large. Proportional action causes the loop output to accelerate toward the setpoint. However by the time the loop response reaches the setpoint value, it has gained inertia from the preceding proportional action. This causes the loop to overshoot the setpoint. As the loop exceeds the setpoint moving toward its first peak,

the error sum is accumulating. This slows down the acceleration, eventually causing the downturn in response.

As the error falls and then drops below the setpoint, the error sum will be reduced because now the error is in the opposite direction. The cycle continues in diminishing peaks until it finally converges at the setpoint as shown in Figure 7-6.

The proportional control action of the loop has a major effect on integral action. Increasing **(PB) Proportional Control Band** results in a smaller integral effect for a given value of **(RP) Reset Period**. In general, decreasing the proportional band, **(PB) Proportional Control Band**, will increase the magnitude of the changes in **(PO) Percent Output**.

Several important factors may not be obvious to inexperienced users of these DDC techniques.

First, whenever the error falls outside of the proportional band - that is,  $\pm(\text{PB})$  **Proportional Control Band**/2 from the setpoint, two important things happen: the controller's output is fully pegged in the appropriate direction, and the error sum stops accumulating. The control produces its maximum output because it must bring the error within the proportional band again. The error sum stops accumulating so that it does not "wind up" a massive error sum that would take many control cycles to dissipate. This feature is called anti-reset windup.

Anti reset windup also makes the loop recover quickly when it reenters the proportional band. Another feature of anti reset windup is that the error history is limited to **(PB) Proportional Control Band**/2 because that is all that required to produce maximum output. Additional error accumulation would only slow down loop recovery.

To quicken loop response while eliminating overshoot, derivative action must be taken. Derivative action takes into account the rate of change in error and allows the GPC to counter the effects of the error's rate of change on the control output. To find the change in error, subtract the current error (read every second by the PID loop) from the previous second's error. A percentage of this change (specified by **(RT) Derivative Rate**) becomes the derivative contribution to the PID output.

The **(RT) Derivative Rate** attribute specifies a percentage of change in error that is to be used in calculating **(PO) Percent Output**. The value is specified in percent per second. The **(RT) Derivative Rate** can have any value from 0.0 to 25.5%/second.

### 7.2.5 SOFT START SETUP

The **(SR) Soft Start Ramp** attribute specifies the maximum percentage change per minute for the associated output under the following conditions: when the controller is initially powered up or reset; upon transitions from unoccupied to occupied mode, upon cancellation of an interlock failure or fire condition, or when a control loop is initially enabled. These situations can cause the control loop to peg to 100% which can cause the output to spike and, in turn, could lead to equipment damage. To prevent this, the output will be limited to changing **(SR) Soft Start Ramp** percent per minute.

### 7.2.6 ENABLING THE CONTROL LOOP

The **(CE) Enable Control Loop** attribute enables and disables the PID loop. When **(CE) Enable Control Loop** = 0, the loop output is not updated but may be set manually. When **(CE) Enable Control Loop** = 1, the loop output is updated by the PID control loop and the corresponding analog output is controlled.

The **(DL) Demand Load** attribute indicates the amount by which **(CS) Calculated Control Setpoint** differs from the loop measured variable.



## 7.3 FLOATING POINT CONTROL

Similar to an Analog PID Control Loop, Floating Point PID Control loops are used to control such devices as fans, pumps, and blowers. Each loop performs either PI or PID control while providing calibration and alarming functions.

In floating point control applications, each floating point control object controls the position of a motor actuator using two digital outputs (an increase output and a decrease output).

### 7.3.1 BASIC SETUP

To initially use a control loop, you must first setup and configure basic properties of the control loop.

#### 7.3.1.1 CONTROL SIGN AND OUTPUT LIMITS

When **(SG) Control Sign** = 0 (normal), a positive error causes an increase in output. When **(SG) Control Sign** = 1 (reverse), a positive error causes a decrease in output. This point determines the response of the loop output to the kind of error. If the output action is to be increased (toward max) when the error is positive, set **(SG) Control Sign** to normal (0). If the output action is to be decreased (toward min) for positive error, set **(SG) Control Sign** to reverse (1).

#### 7.3.1.2 MEASURED VARIABLE CONFIGURATION

Attributes **(IC) Input Channel** and **(IA) Input Attribute** specify the channel and attribute to be used as the loop measured variable. It specifies the input to be used for the control loop's measured variable. Any data point within the GPC can be used as the measured variable by configuring these properties appropriately.

When the control loop is enabled, **(IV) Input's Present Value** will reflect the current value of the loop measured variable.

#### 7.3.1.3 OUTPUT PAIR SELECTION

Attribute **(DO) DO Pair Selection** selects digital output pairs used by the GPC to perform motor control. DOs are grouped by their logical order.

#### 7.3.1.4 SETPOINT CONFIGURATION

Each control loop contains a **(SP) Loop Setpoint** attribute, defining the control setpoint that is used for a specific schedule mode. The setpoint is expressed in the same kind of measurement units (engineering units) that the measured variable uses (e.g., degrees, cfm, inches of WC, etc.).

The **(SU) Unoccupied Setup/Setback** attribute allows the calculated setpoint to be setback during unoccupied periods.

The **(P0) Permanent Stat Override Enabled** attribute allows the user to select whether a setpoint override made from a STAT should be permanent. When enabled, the override is made permanent and persists through controller reboots. Corresponding with the addition of this new variable, the **(TS) Thermostat SP Adjustment** will only effect the setpoint when the controller is in an occupied or warmup schedule mode.

When the control loop is later enabled (using **(CE) Enable Control Loop**), the **(CS) Calculated Control Setpoint** will reflect which setpoint is currently active.

#### 7.3.1.5 SCHEDULES TO FOLLOW

The **(SM) Schedules to Follow** attribute allows control loop to reference a schedule and transition through programmed set points appropriately. Each bit in **(SM) Schedules to Follow** corresponds to any of the schedule channels within the GPC, as well as STAT Overrides, and other options. These bits are summarized in Table 7-2.

Table 7-2 : Schedules to Follow

SM bit	Schedule
0	F901
1	F902
2	F902
3	F903 4
4	F904
5	F905
6	F906
7	F907
8	F908
9 - 21	STAT Override UIx
22	Host Schedule (F900;HO)
23	F900;CV
24	Occupancy Extension (FC01)

If you do not want to use schedule control, set all of the bits in **(SM) Schedules to Follow** to 0.

### 7.3.2 PROPORTIONAL CONTROL SETUP

The **(PB) Proportional Control Band** specifies the input variable range over which the output value is proportional to the error value (i.e., changes in the measured variable result in proportional changes in the output signal). The proportional band is centered around setpoint for the loop. This point is expressed in the same kind of measurement units (engineering units) that the measured variable uses—for example: degrees, cfm, inches of WC.

To determine **(PB) Proportional Control Band**, first decide how closely the GPC must control to the setpoint. For instance, if the setpoint is 72°F, then an acceptable control range might be within two degrees of the setpoint. This control range can be expressed as a band centered on the setpoint: from 70° to 74°, or 4 degrees—the *proportional band*. Refer to Figure 7-1 and Figure 7-2.

For normal acting control loops (see Figure 7-1), the **(PO) Percent Output** attribute is set to maximum output when the input variable equals the setpoint plus half of the proportional band (**(CS) Calculated Control Setpoint + (PB) Proportional Control Band/2**). The percent output is set to minimum output when the input variable equals the setpoint minus half of the proportional band (**(CS) Calculated Control Setpoint - (PB) Proportional Control Band/2**). These associations are reversed for reverse acting control loops. **(PO) Percent Output** will be midway between minimum and maximum output when the measured variable is equal to the control setpoint **(CS) Calculated Control Setpoint**. The opposite would be true for reverse acting control loop as shown in Figure 7-2.

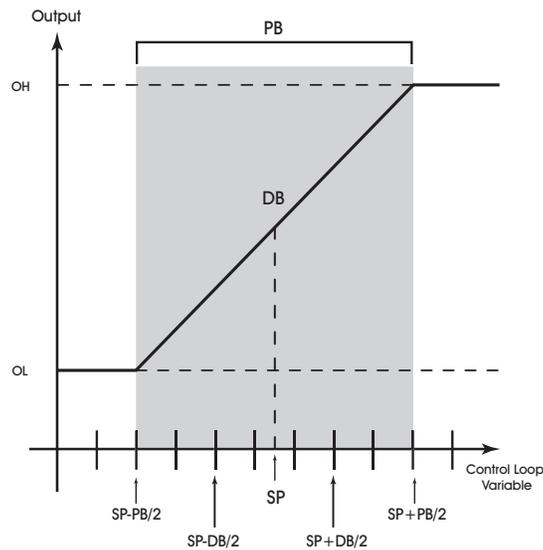


Figure 7-7: Proportional Band for Normal Acting Control ( $SG = 0$ )

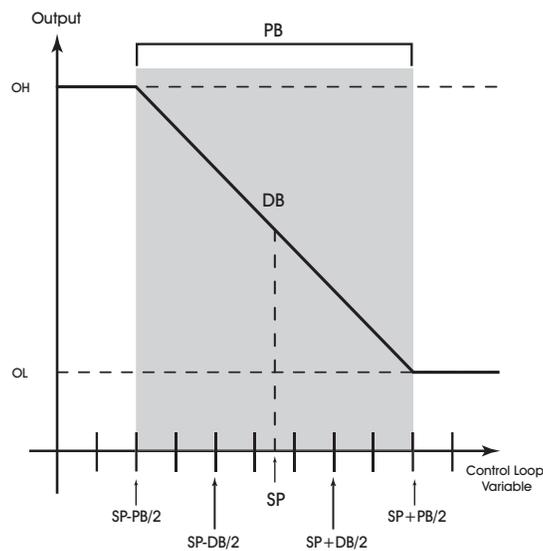


Figure 7-8: Proportional Band for Reverse Acting Control ( $SG = 1$ )

Proportional only control produces cycling, and its performance changes when the measured environment changes. The way to eliminate cycling and to compensate for load changes is to use *integral* action, the “I” part for PID control.

### 7.3.3 DEADBAND CONFIGURATION

The **(DB) Desired Control Deadband** attribute specifies the deadband within the proportional control band in which the output remains constant at a point midway between maximum output and minimum output. By specifying a deadband that is greater than or equal to the resolution of the loop measured variable, you eliminate the possibility of cycling around the setpoint. The value of the deadband should

never exceed the proportional band. If the deadband is greater than the proportional band, then the control loop will not have proportional control.

The deadband is used to specify an input variable range within the proportional band. The size of the deadband should be based on the loop measured variable. When the value of the measured variable is within this dead band, the output signal remains constant at the midpoint of the minimum/maximum range.

The point that deadband is centered on one of the four defined set points to create the actual control dead band. When the value of the loop measured variable is within  $\pm(\text{DB})$  **Desired Control Deadband/2** of the setpoint, the GPC assumes that it has reached the setpoint. Refer to Figure 7-3.

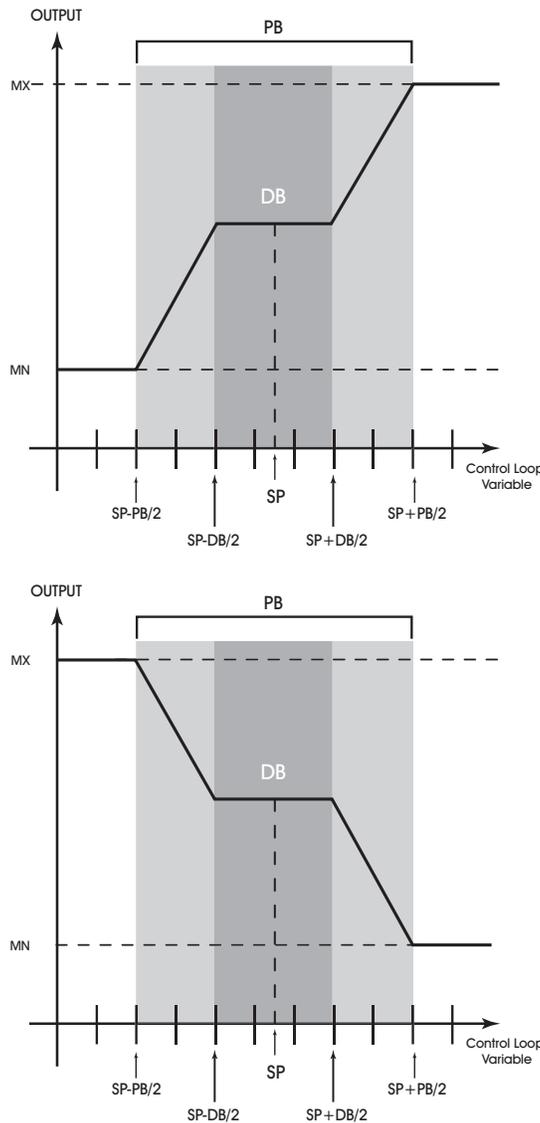


Figure 7-9: Normal Acting (above) and Reverse Acting (below), Proportional Control Output Response Showing a Dead Band Centered Around the Setpoint (SP)

By entering a value in deadband that is greater than the resolution of the measured variable sensor, you create a deadband that allows the GPC to effectively reach setpoint. Be sure that the deadband selected does not exceed the size of the proportional band.

**CAUTION**

*Never change **DB** to a value greater than half of the proportional band **PB**. Doing so will eliminate the effects of PID control, resulting in on/off control.*

#### 7.3.4 RESET CONTROL SETUP

The **(MR) Maximum Amount to Reset Setpoint** attribute specifies the maximum amount to reset the **(SP) Loop Setpoint** based on when reset is being used. The **(CS) Calculated Control Setpoint** takes into account the use of the maximum reset specified in **(MR) Maximum Amount to Reset Setpoint**.

Attributes **(RC) Reset Variable** and **(RA) Reset Attribute** specify the channel and attribute to be used as the Reset Variable.

The **(RS) Reset Setpoint** attribute specifies the value at which the reset action begins. When the value of the reset variable exceeds **(RS) Reset Setpoint**, reset action will be used in determining the calculated setpoint. Just as **SP** is the proportional control setpoint for the measured variable specified in **IC** and **IA**, **RS** is the reset control setpoint for the value of the reset variable selected by **(RC) Reset Variable** and **(RA) Reset Attribute**.

The **(RL) Reset Limit** attribute specifies the value at which maximum reset is used. When the value of the reset variable is equal to **(RL) Reset Limit**, the **(MR) Maximum Amount to Reset Setpoint** is used in determining the calculated setpoint (**CS**).

The relationship between **(RL) Reset Limit** and **(RS) Reset Setpoint**, as well as the sign (+ or -) of **MR**, determines how changes in the reset variable specified by **(RC) Reset Variable** and **(RA) Reset Attribute** affect the calculated control setpoint (**CS) Calculated Control Setpoint**. Refer to Figure 7-4. In the illustrations, references to **(SP) Loop Setpoint** generically refer to your currently used scheduled setpoint.

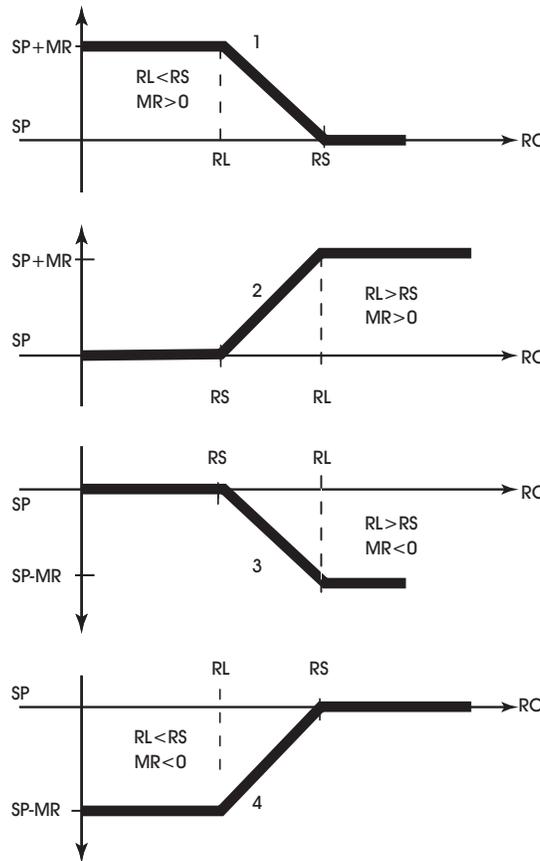


Figure 7-10: Four Forms of Reset Action

With appropriate values entered for these properties, the GPC will provide simple closed loop feedback proportional control. This means that the actual measured performance of the control (from the measured variable input) is fed back to the controller and is compared with the effective setpoint for the loop. Any difference between the actual value of the measured variable and effective setpoint values is called error.

One problem with proportional only control is the changes in loop performance that occur when the condition being measured by the input sensor changes (e.g., the measured temperature changes when a door is opened and the room or space is flooded with cold air). As the loop environment changes, the proportional only control loop begins to cycle around an offset from the setpoint. Figure 7-5 illustrates the performance of a typical loop under proportional only control.

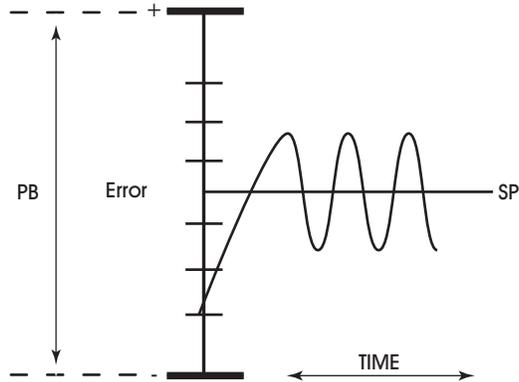


Figure 7-11: Proportional Only Control

Rather than responding exclusively to the loop error from moment to moment as is the case with proportional action, integral action is based on a summation of the error that has occurred over some period. This error sum is used to reset, or modify, the response of the control loop (output) based on a running average of the error. The amount of time over which the error averaging is accumulated is called the *reset period*.

The **(RP) Reset Period** attribute specifies the reset period (in seconds) over which the error history is accumulated. If **(RP) Reset Period** = 10 seconds with a constant error of 2.0, then the error history would increase by 0.2 every second. In five seconds, the error history would be 1.0. At the end of ten seconds, the error history would be 2.0. Setting **(RP) Reset Period** to 0 disables integral action making the loop proportional only. The longer **(RP) Reset Period** is, the less effect it has on the control response. Figure 7-6 shows the response of a typical control loop when integral action is used in addition to proportional action (PI control). A value of 0 disables the reset period.

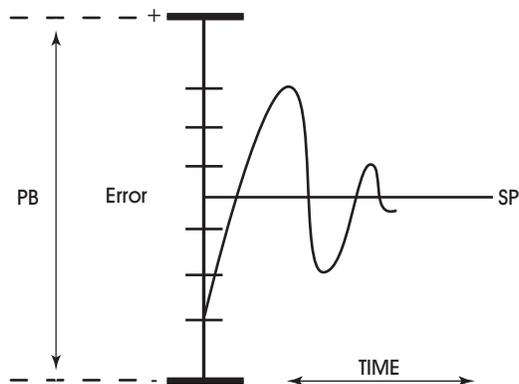


Figure 7-12: Proportional + Integral (PI) Control

At the start-up of the loop or following a change in setpoint (see Figure 7-6), the error is fairly large. Proportional action causes the loop output to accelerate toward the setpoint. However by the time the loop response reaches the setpoint value, it has gained inertia from the preceding proportional action. This causes the loop to overshoot the setpoint. As the loop exceeds the setpoint moving toward its first peak,

the error sum is accumulating. This slows down the acceleration, eventually causing the downturn in response.

As the error falls and then drops below the setpoint, the error sum will be reduced because now the error is in the opposite direction. The cycle continues in diminishing peaks until it finally converges at the setpoint as shown in Figure 7-6.

The proportional control action of the loop has a major effect on integral action. Increasing the **(PB) Proportional Control Band** results in a smaller integral effect for a given value of **(RP) Reset Period**. In general, decreasing the proportional band, **(PB) Proportional Control Band**, will increase the magnitude of the changes in **(PO) Percent Output**.

Several important factors may not be obvious to inexperienced users of these DDC techniques.

First, whenever the error falls outside of the proportional band—that is,  $\pm(\text{PO}) \text{ Percent Output}/2$  from the setpoint, two important things happen: the controller's output is fully pegged in the appropriate direction, and the error sum stops accumulating. The control produces its maximum output because it must bring the error within the proportional band again. The error sum stops accumulating so that it does not “wind up” a massive error sum that would take many control cycles to dissipate. This feature is called anti reset windup.

Anti reset windup also makes the loop recover quickly when it reenters the proportional band. Another feature of anti reset windup is that the error history is limited to **(PO) Percent Output/2** because that is all that required to produce maximum output. Additional error accumulation would only slow down loop recovery.

To quicken loop response while eliminating overshoot, derivative action must be taken. Derivative action takes into account the rate of change in error and allows the GPC to counter the effects of the error's rate of change on the control output. To find the change in error, subtract the current error (read every second by the PID loop) from the previous second's error. A percentage of this change (specified by **(RT) Derivative Rate**) becomes the derivative contribution to the PID output.

The **(RT) Derivative Rate** attribute specifies a percentage of change in error that is to be used in calculating **PO**. The value is specified in percent per second. The **(RT) Derivative Rate** can have any value from 0.0 to 25.5%/second.

### 7.3.5 CALIBRATION

The **(TT) Motor Travel Time** attribute is used to specify the total time in seconds (0 to 65,535 seconds) that it takes the motor actuator to go full stroke (from fully open to fully closed). **(TT) Motor Travel Time** is used to determine the **(CP) Current Position** of the motor. **(TT) Motor Travel Time** defaults to a value of 0 seconds.

## CAUTION



The travel time of a motor depends on the load that is applied to the motor. For accuracy, it is suggested that you determine **(TT) Motor Travel Time** when the motor is loaded. For spring loaded motors, the full stroke travel time from 0% to 100% may be different than the 100% to 0% travel time. You may choose to use the higher of the two travel times for **(TT) Motor Travel Time**. In this case, it is recommended that you perform regular calibrations on the motor.

The actuator can be manually calibrated by enabling floating point control pair enable (**PE=1**), disabling PI control (**CE=0**) and setting **DP** to 0% or 100%. When the actuator is at the programmed position (after approximately **TT** seconds), set **CP** to 0% or 100% accordingly. Finally, be sure to return PI control (**CE=1**) if you want **DP** to be set automatically.

Floating point control loops can be calibrated automatically by the GPC at programmable intervals. This is done using the recalibrate interval. The **(RI) Motor Recalibration Interval** attribute specifies how often (if at all) the associated floating point control object is to be recalibrated.

**(RI) Motor Recalibration Interval** is given in hours (0-255 hours). If **(RI) Motor Recalibration Interval=0**, then recalibration of floating point control loops does not occur. If **(RI) Motor Recalibration Interval>0**, recalibration of the associated floating point control loops occurs every **(RI) Motor Recalibration Interval** hours.

The GPC recalibrates the floating point control loops by driving the **(DP) Desired Position** to the fully closed position (0%) for the amount of time specified in the **(TT) Motor Travel Time**. The GPC then sets the current position to 0%, after which the recalibration is complete and the controller returns the desired position to its original value.

For floating point control objects, you can enable an automatic creep feature using **(CR) Motor Creep Function**. This feature is used to automatically calibrate the output when its desired position is either 0% or 100%. The automatic creep feature is performed in one of two ways: (1) the appropriate output is left on when the output signal is at 0% or 100%, or (2) the output is *creeped* (pulsed) at a rate of 1% per minute (the current position is set to 1% or 99%) when the output signal is at 0% or 100%.

These two methods of output correction (continuous on and automatic creep) are illustrated in Figure 7-13. This example shows a floating point control loop with a desired position of 100%.

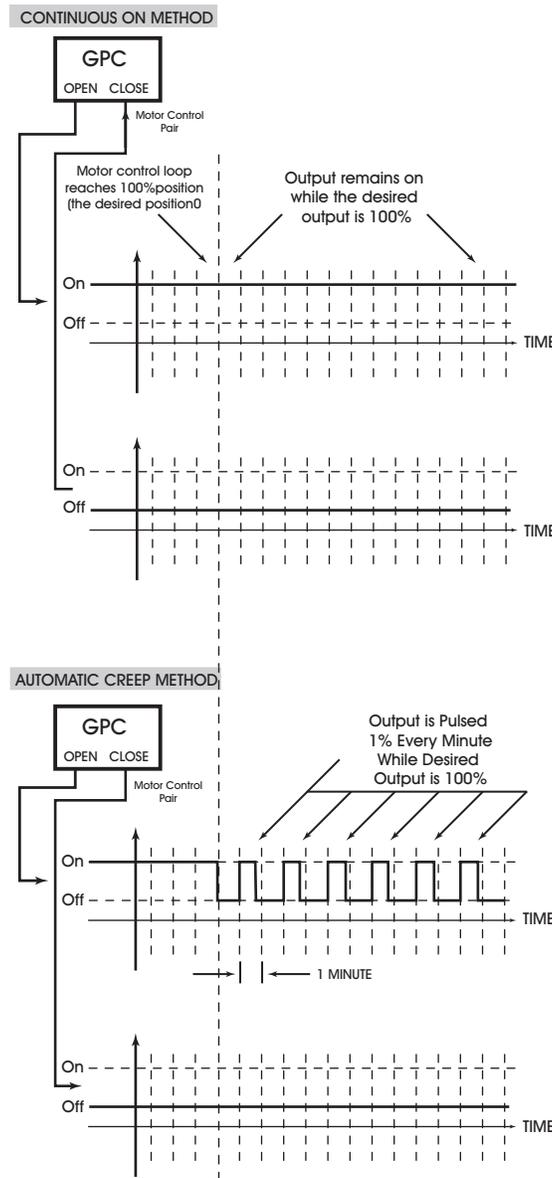


Figure 7-13 Auto Creep

### 7.3.6 INTERLOCKING AND FIRE POSITIONING

Input Interlocking permits the control loop to be set to a default position in the event that an Universal Input (configured as a Digital Input) has a positive value.

To configure Interlocking, select one or more inputs from **(IL) Inputs for Interlocking**. When multiple inputs are selected, the GPC operates in an **OR** fashion, where if any of the selected inputs contain a positive value, the loop will be forced to a specific position.

The position that the loop is commanded to is dictated through attribute **(FP) Interlock/Comm Failure Position**

For Fire Mode situations, **(FI) Fire Position** defaults the control loop to a specific value.

### 7.3.7 ENABLING THE CONTROL LOOP

The **(CE) Enable Control Loop** attribute enables and disables the loop. When **(CE) Enable Control Loop** = 0, the loop output is not updated but may be set manually. When **(CE) Enable Control Loop** = 1, the loop output is updated by the PID control loop and the corresponding analog output is controlled.

If the **(DP) Desired Position** of the motor is greater than the current position, the controller will drive the motor open by turning on the “increase” output for a calculated period of time. If the desired position is less than the current position, the controller will drive the motor closed by turning on the “decrease” output for a calculated period of time.

The desired position of the floating point control object can be set manually or calculated automatically by the PI algorithm. The automatic floating point control algorithm operates as follows. When the value of the selected measured variable is within the control loop’s deadband, no control action is taken by the PI loop. When the value of the measured variable is outside the deadband, but within a programmable proportional band, the output is modulated using PI control according to the setpoint of the control loop. When the value of the measured variable is outside the deadband and beyond (either above or below) the proportional band, the output is set to either 0% or 100%, as appropriate.

## 7.4 THERMOSTATIC CONTROL

Thermostatic Control Loops are used to provide effective on/off binary control. When thermostatic control is enabled, the present-value can be used to control binary outputs. By calculating a control setpoint and comparing it with the measured variable, the loop can determine the output value necessary to maintain the desired setpoint. The control loop can enforce a control deadband to prevent hysteresis and can be configured to operate based on one or multiple pre-defined schedule to allow the loop to differentiate setpoint control.

### 7.4.1 BASIC SETUP

To initially use a control loop, you must first setup and configure basic properties of the control loop.

#### 7.4.1.1 CONTROL ACTION

The **(SG) Control Action** defines the control sign of the thermostatic control loop, where 0 = Cooling, and 1 = Heating.

#### 7.4.1.2 MEASURED VARIABLE CONFIGURATION

The **(IC) Input Channel** and **(IA) Input Attribute** specify the channel and attribute to be used as the loop measured variable. It specifies the input to be used for the control loop's measured variable. Any valid set within the GPC can be used as the measured variable by configuring these properties appropriately.

#### 7.4.1.3 SETPOINT CONFIGURATION

Each control loop contains a **(SP) Loop Setpoint** attribute, defining the control setpoint that is used for a specific schedule mode. The setpoint is expressed in the same kind of measurement units (engineering units) that the measured variable uses (e.g., degrees, cfm, inches of WC, etc.).

The **(SU) Unoccupied Setup/Setback** attribute allows the calculated setpoint to be setback during unoccupied periods.

The **(P0) Permanent Stat Override Enabled** attribute allows the user to select whether a setpoint override made from a STAT should be permanent. When enabled, the override is made permanent and persists through controller reboots. Corresponding with the addition of this new variable, the **(TS) Thermostat SP Adjustment** will only effect the setpoint when the controller is in an occupied or warmup schedule mode.

When the control loop is later enabled (using **(CE) Enable Control Loop**), the **(CS) Calculated Control Setpoint** will reflect which setpoint is currently active.

#### 7.4.1.4 SCHEDULES TO FOLLOW

The **(SM) Schedules to Follow** attribute allows control loop to reference a schedule and transition through programmed set points appropriately. Each bit in **(SM) Schedules to Follow** corresponds to any of the schedule channels within the GPC, as well as STAT Overrides, and other options. These bits are summarized in Table 7-2.

Table 7-3 : Schedules to Follow

SM bit	Schedule
0	F901
1	F902
2	F902

Table 7-3 : Schedules to Follow

SM bit	Schedule
3	F903 4
4	F904
5	F905
6	F906
7	F907
8	F908
9 - 21	STAT Override UIx
22	Host Schedule (F900;HO)
23	F900;CV
24	Occupancy Extension (FC01)

If you do not want to use schedule control, set all of the bits in **(SM) Schedules to Follow** to 0.

#### 7.4.1.5 CONFIGURING LOOP PARAMETERS

The **(DB) Desired Control DeadBand** attribute specifies a control deadband for the thermostatic control loop. For a normal action control, this specifies the amount by which the temperature must drop below the cooling setpoint before the output is de-energized (**(SP) Loop Setpoint-(DB) Desired Control DeadBand**). For a reverse action control, this specifies the amount by which the temperature must rise above the heating setpoint before the output is de-energized (**(SP) Loop Setpoint+(DB) Desired Control DeadBand**). This response is illustrated in Figure 7-14.

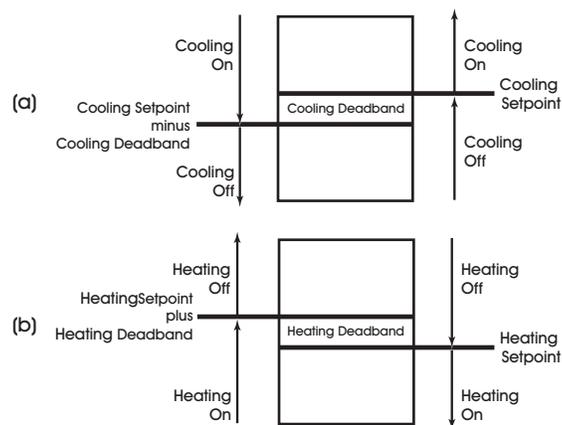


Figure 7-14: Deadband for a Normal Acting (a) and Reverse Acting (b) Thermostatic Control Loop

#### 7.4.2 ENABLING THE CONTROL LOOP

The **(CE) Enable Control Loop** attribute enables and disables the loop. When **(CE) Enable Control Loop** = 0, the loop output is not updated but may be set manually. When **CE** = 1, the loop output is updated by the PID control loop and the corresponding analog output is controlled.

Once enabled, the thermostatic control object will control based on the **(CS) Calculated Control Setpoint** attribute. This attribute represents the desired temperature in the area being controlled. The controller will begin with the setpoint value based on your current schedule mode (if schedules have been selected).

The value of **(CS) Calculated Control Setpoint** is compared to the measured variable. The difference between **(CS) Calculated Control Setpoint** and the measured variable will be stored in the **(DL) Demand Load** attribute. If the measured variable does not equal the calculated setpoint and is outside of the specified control deadband, then action will be taken to correct the measured variable.

The **(CV) Current Value** attribute indicates the current state of the control loop. Control loop conditions are Off or On.

---

## SECTION 8: SCHEDULING

*This section describes the process of setup and configuration for the GPC's scheduling system, including host scheduling, broadcast scheduling, occupancy extension, local schedules and holiday concepts.*

### IN THIS SECTION

Scheduling Overview .....	8-3
Schedule Summary.....	8-3
Main Schedule .....	8-4
Host Schedule.....	8-5
Broadcast Schedule.....	8-6
Occupancy Extension .....	8-7
Local Schedules.....	8-8
Holidays .....	8-12



## 8.1 SCHEDULING OVERVIEW

The GPC supports traditional time-based scheduling using AAM's classic four-mode scheduling routine. Each GPC supports up to 8 defines schedules.

### 8.1.1 SCHEDULE SUMMARY

The Schedule Summary provides an overview of all eight (8) local schedules within the GPC via attributes **C1** through **C8**, as well as the current Host Schedule status, Holiday Status, and STAT Overrides.

System		Inputs	Outputs	STATbus	Programs	Control Loops	Scheduling	Data Manipulation	Data Storage And Movement	Special Mod
Schedule Summary		Main Schedule	Host Schedule	Broadcast Schedule	Occupancy Extension	Local Schedules	Holidays			
F900 ON	✓	Schedule Summary Name	<input type="text" value="Schedule Summary"/>							
F900 CV	✓	Main Schedule Current Value	<input checked="" type="radio"/> Unoccupied	<input type="radio"/> Warmup	<input type="radio"/> Occupied	<input type="radio"/> Night Setback				
F900 C1	✓	Schedule 1	<input checked="" type="radio"/> Unoccupied	<input type="radio"/> Warmup	<input type="radio"/> Occupied	<input type="radio"/> Night Setback				
F900 C2	✓	Schedule 2	<input checked="" type="radio"/> Unoccupied	<input type="radio"/> Warmup	<input type="radio"/> Occupied	<input type="radio"/> Night Setback				
F900 C3	✓	Schedule 3	<input checked="" type="radio"/> Unoccupied	<input type="radio"/> Warmup	<input type="radio"/> Occupied	<input type="radio"/> Night Setback				
F900 C4	✓	Schedule 4	<input checked="" type="radio"/> Unoccupied	<input type="radio"/> Warmup	<input type="radio"/> Occupied	<input type="radio"/> Night Setback				
F900 C5	✓	Schedule 5	<input checked="" type="radio"/> Unoccupied	<input type="radio"/> Warmup	<input type="radio"/> Occupied	<input type="radio"/> Night Setback				
F900 C6	✓	Schedule 6	<input checked="" type="radio"/> Unoccupied	<input type="radio"/> Warmup	<input type="radio"/> Occupied	<input type="radio"/> Night Setback				
F900 C7	✓	Schedule 7	<input checked="" type="radio"/> Unoccupied	<input type="radio"/> Warmup	<input type="radio"/> Occupied	<input type="radio"/> Night Setback				
F900 C8	✓	Schedule 8	<input checked="" type="radio"/> Unoccupied	<input type="radio"/> Warmup	<input type="radio"/> Occupied	<input type="radio"/> Night Setback				
F900 HO	✓	Host Schedule Value	<input checked="" type="radio"/> Unoccupied	<input type="radio"/> Warmup	<input type="radio"/> Occupied	<input type="radio"/> Night Setback				
F900 HE	✓	Enable Host Schedule?	<input checked="" type="radio"/> No	<input type="radio"/> Yes						
F900 DH	✓	Today's Holiday Status	<input checked="" type="radio"/> No	<input type="radio"/> Yes						
F900 TH	✓	Tomorrow's Holiday Status	<input checked="" type="radio"/> No	<input type="radio"/> Yes						
F900 IS	✓	Inactive state (All Schedules)	<input checked="" type="radio"/> Unoccupied	<input type="radio"/> Warmup	<input type="radio"/> Occupied	<input type="radio"/> Night Setback				
F900 AS	✓	Active Schedules	<input type="checkbox"/> F901	<input type="checkbox"/> F902	<input type="checkbox"/> F903	<input type="checkbox"/> F904	<input type="checkbox"/> F905	<input type="checkbox"/> F906	<input type="checkbox"/> F907	<input type="checkbox"/> F908
F900 SO	✓	Digital Stat Override Status	<input type="checkbox"/> UI 9	<input type="checkbox"/> UI 10	<input type="checkbox"/> UI 11	<input type="checkbox"/> UI 12				

Figure 8-1 : Schedule Summary

## 8.2 MAIN SCHEDULE

The Main Schedule area provides information on the Schedule's **(CV) Current Value**, as well as the method being used to determine the current schedule state via **(FB) Main Schedule Set By** attribute.

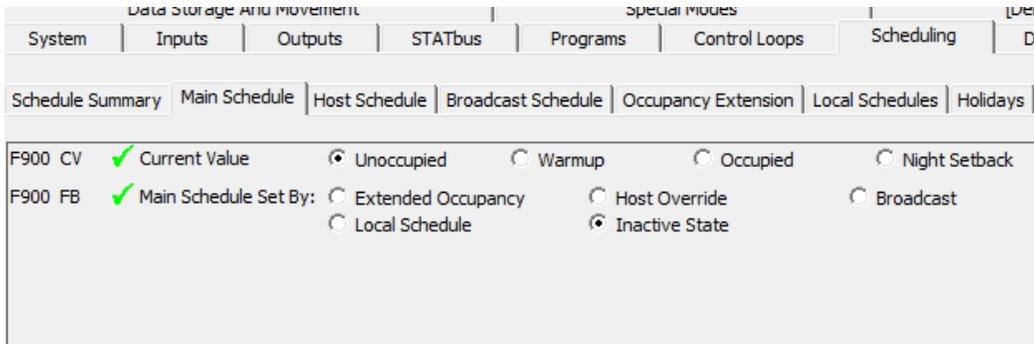


Figure 8-2 : Main Schedule Area

### 8.3 HOST SCHEDULE

The GPC is capable of having its schedule status dictated by a connected host or area controller solution through the Host Schedule routine. To enable Host Scheduling, set **(HE) Enable Host Schedule = 1 (Yes)**. When enabled, schedules status will be dictated exclusively through attribute **(HO) Host Schedule Value**. All local schedules will be ignored.

In this scenario, a connected host or area controller solution must write the current schedule status to attribute **(HO) Host Schedule Value**.

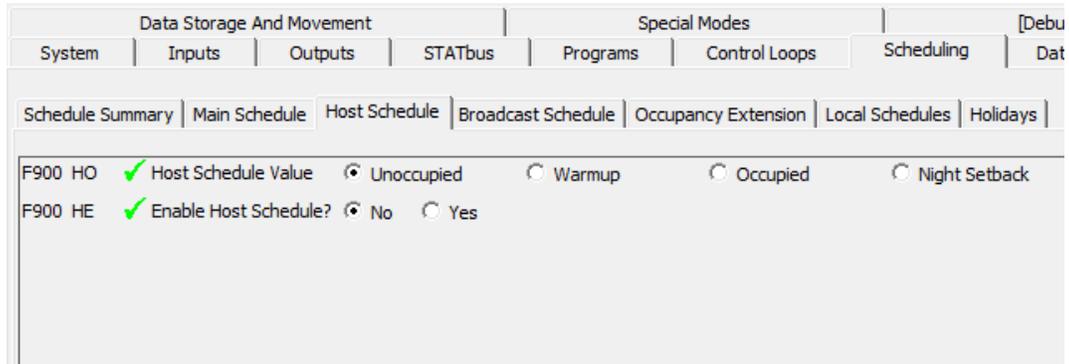


Figure 8-3 Host Schedule

## 8.4 BROADCAST SCHEDULE

Scheduling can also be interpreted through broadcast. GPC has the capability to receive a schedule value. Additionally, it may also transmit a schedule broadcast to locally connected PUP devices, allowing them to be controlled off the GPC's configured schedule.

Attribute	Value
F005 ON	✓ Broadcast Name: Broadcast Schedule
F005 CV	✓ Current Value: 0
F005 DT	✓ PUP Datatype for CV: 254
F005 BM	✓ Broadcast Mode: <input checked="" type="radio"/> 0 = Off <input type="radio"/> 1 = Send <input type="radio"/> 2 = Receive
F005 BZ	✓ Broadcast Type: <input type="radio"/> Zone Broadcast <input checked="" type="radio"/> Global Broadcast
F005 ZN	✓ Zone Number: 0
F005 IC	✓ Input Channel: F900
F005 IA	✓ Input Attribute: CV
F005 BT	✓ Minutes Between Broadcasts: 5
F005 ES	✓ Elapsed Seconds Since Broadcast: 65535
F005 FB	✓ Feedback Status: <input type="radio"/> Set To Receive <input type="radio"/> Unable To Read Input <input checked="" type="radio"/> Working Properly <input type="radio"/> Unable To Coerce Input Datatype

First, you must determine whether or not you wish to send or receive a Schedule value. This is determined through attribute **(BM) Broadcast Mode**. By default, GPC will not send or receive schedule broadcasts.

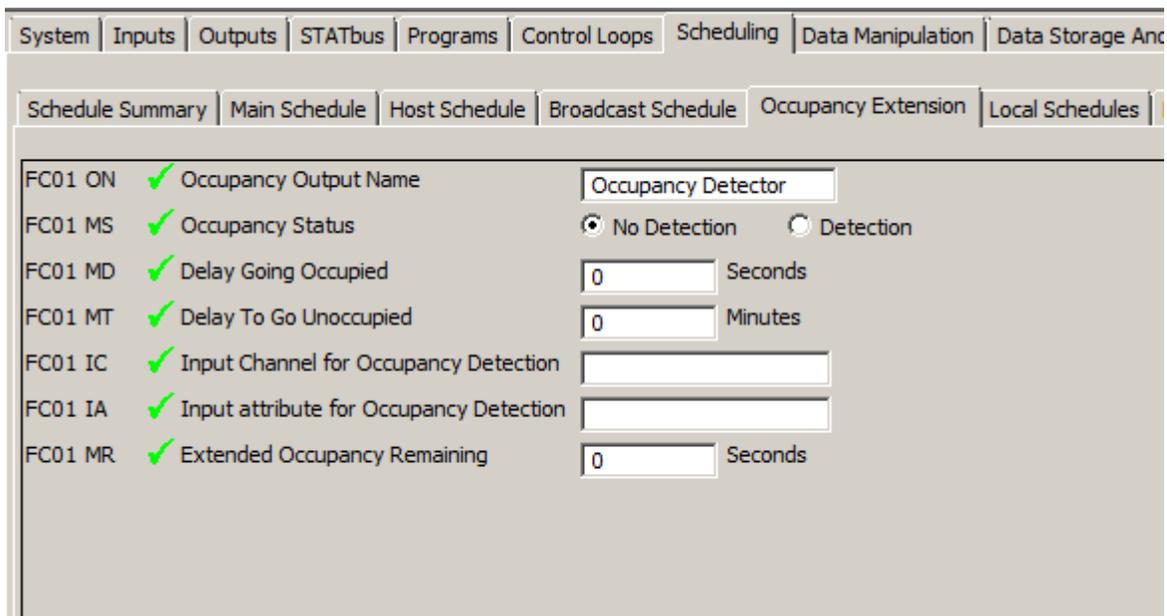
If you have elected to receive a Schedule broadcast, simply assure that **(ZN) Zone Number** is configured to accept the broadcast if it is being sent as a Zone Broadcast. If a unit sending a Schedule broadcast sends the message globally, **(ZN) Zone Number** needs no configuration.

If you have elected to have the GPC send broadcasts, your GPC must either be configured as a Full Administrator or be passed a network token. Set **(BM) Broadcast Mode** to a value of 2 (Receive). **(BT) Minutes Between Broadcasts** defines how often, in minutes, a broadcast message will be sent by the GPC. Attributes **(ES) Elapsed Seconds Since Broadcast**, and **(FB) Feedback Status** provide diagnostic indication regarding the current status of the Schedule broadcast.

## 8.5 OCCUPANCY EXTENSION

The Occupancy Extension feature of the GPC permits the ability to perform extended occupancy duration for schedule override situations. Attributes **(IC) Input Channel for Occupancy Detection** and **(IA) Input Attribute for Occupancy Detection** must be defined in order to determine the point designation that is monitored for detection. **(MD) Delay going Occupied** specified the delay in seconds where detection must remain positive before Occupancy status changes. Finally, **(MT) Delay to go Unoccupied** must be set to define how long, in minutes, Occupancy occurs when detected.

When detection occurs, attribute **(MR) Extended Occupancy Remaining** displays how much time, in seconds, remains for a triggered Occupancy. This counter will reset back to the value specified in **(MT)** if the input is re-triggered. or if the input remains active, the time remaining will not count down.



Attribute	Status	Value	Unit
FC01 ON	✓	Occupancy Output Name	Occupancy Detector
FC01 MS	✓	Occupancy Status	<input checked="" type="radio"/> No Detection <input type="radio"/> Detection
FC01 MD	✓	Delay Going Occupied	0 Seconds
FC01 MT	✓	Delay To Go Unoccupied	0 Minutes
FC01 IC	✓	Input Channel for Occupancy Detection	
FC01 IA	✓	Input attribute for Occupancy Detection	
FC01 MR	✓	Extended Occupancy Remaining	0 Seconds

Figure 8-4 : Occupancy Extension

## 8.6 LOCAL SCHEDULES

The GPC supports a total of 8 local schedules - used to define and perform time of day occupancy scheduling within the controller. Each controller has four attributes defining the start of each schedule transition period - including **(WO) Time to go Warm-Up**, **(OC) Time to go Occupied**, **(NS) Time to go Into Night Setback**, and **(UN) Time to go Unoccupied**. Effectivity of the schedule for specific days is achieved through configuration of the **(AD) Active Days** attribute.

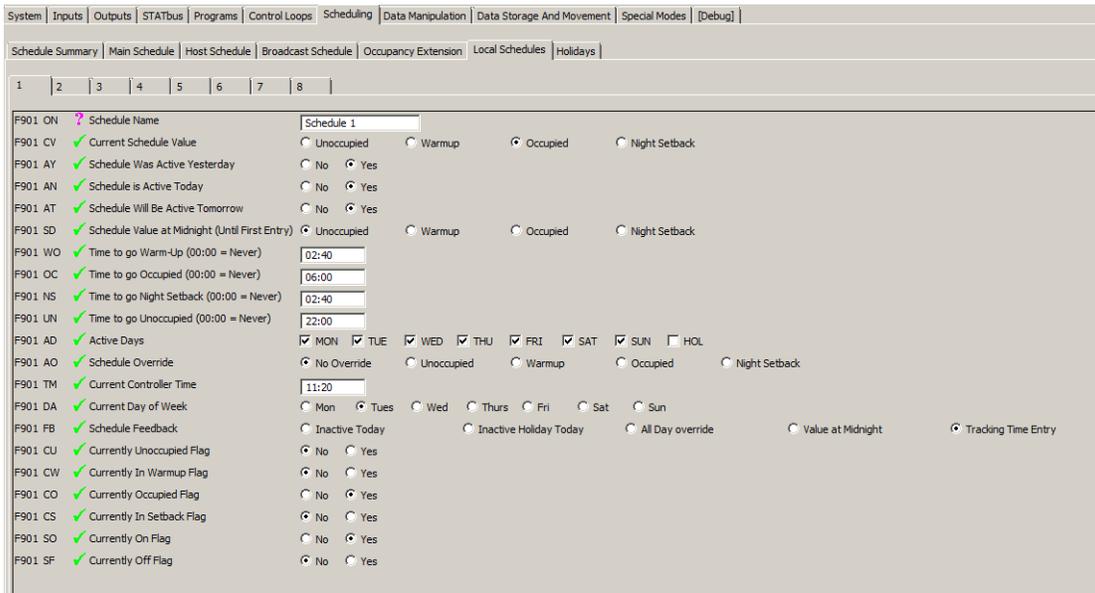


Figure 8-5 : Local Schedules

### 8.6.1 OPERATION

The SBC-GPC operates in one of four scheduled control states: occupied, unoccupied, warm-up and night setback.

- Unoccupied mode, **(CV) Current Value=0**, is the period of time when people are not expected to be in the zone and temperature control is not as strict. During unoccupied mode, the SBC-GPC maintains cooling comfort levels at setup values and heating comfort levels at setback values. These setup and setback values are used to broaden the control range between the heating and cooling setpoints in order to provide less stringent control. The attributes used to define the offsets are located in the individual control loop channels, but the time of implementation is set in the individual schedules F901-F908. Unoccupied mode usually ends when night setback begins.
- Warmup, **(CV) Current Value=1**, is the period of time before occupancy. During this period, the central air handler unit supplies warm air to the VAV boxes. Warmup provides special control action to bring the zone temperature to its desired setpoint for the occupied mode, based on the heating setpoint. The warmup period ends when occupied mode begins.
- Occupied mode, **(CV) Current Value=2**, is the period of time when the zone is occupied by people and the SBC-GPC must maintain appropriate comfort levels in the zone. The heating and cooling setpoints define a desired zone temperature range. Occupied mode ends when unoccupied mode time begins.
- Night setback, **(CV) Current Value=3**, is the period of time during unoccupied mode when the entire building is usually unoccupied and the air handler may be shut down. The SBC-GPC provides the option to set up and set back the night setback control temperature (as does the standard unoccupied mode), and when these offsets are reached or exceeded damper control of air flow resumes.

SBC-GPC schedules can be activated based on the values assigned to attributes in the Schedule channels F901-F908. When the current day of the week matches the setting of the active days (**AD**) attribute from one of the eight schedule channels, that channel's schedule becomes active. More than one schedule can be active at any given time, but they are prioritized so that the schedule with the highest mode priority dictates the control mode. Priority is determined in the following order:

- . occupied (highest priority)
- . warmup
- . night setback
- . unoccupied (lowest priority)

The schedule mode attributes define four windows for a schedule which is active for a set of days of the week. When the current day of the week matches one of the active schedule days the time of day determines which of the four available modes that will dictate control strategy.

Scheduling can also be accomplished via the (**SD**) **Schedule Value at Midnight (Until First Entry)**. On active days, the schedule's (CV) Current Value is set to this value at midnight. If a different value is desired at midnight, then it should be entered as going into effect at 00:00. This value defaults to "Unoccupied" but can be configured in the event that the controller is installed in a location where the zone is not supposed to be unoccupied at midnight every night.

The active days for the schedule are designated by the (**AD**) **Active Days** attribute. (**AD**) **Active Days** specifies a set of the eight possible days of the week (seven days plus holiday) during which the schedule will run in one of the four available modes at any given time of that particular active day.

Attribute (**AD**) **Active Days** is used to specify the active days for the schedule. The active days are those day for which the schedule will be applied. (**AD**) **Active Days** is a bitmap with bit #0 corresponding to Monday, bit #1 corresponding to Tuesday, etc. up to bit #6 which corresponds to Sunday. Bit #7 of (**AD**) **Active Days** corresponds to a Holiday. Setting a bit of (**AD**) **Active Days** equal to 1 makes the schedule active on the corresponding day.

Attribute (**AO**) **All Day Override** is used to set the schedule state for the entire day. This can be used, for example, for holidays when you wish to keep the schedule unoccupied all day long. (**AO**) **All Day Override** would also be used whenever you wished for the schedule to stay in a certain state around the clock without interruption. Figure 8-6 shows an examples of schedule.

Table 8-1 :Options for AO

AO	Schedule
0	None
1	Unoccupied
2	Warmup
3	Occupied
4	Night Setback

8.6.2 SCHEDULE EXAMPLE

The following graphic represents a general schedule and how the SBC-GPC would be configured to carry out this process.

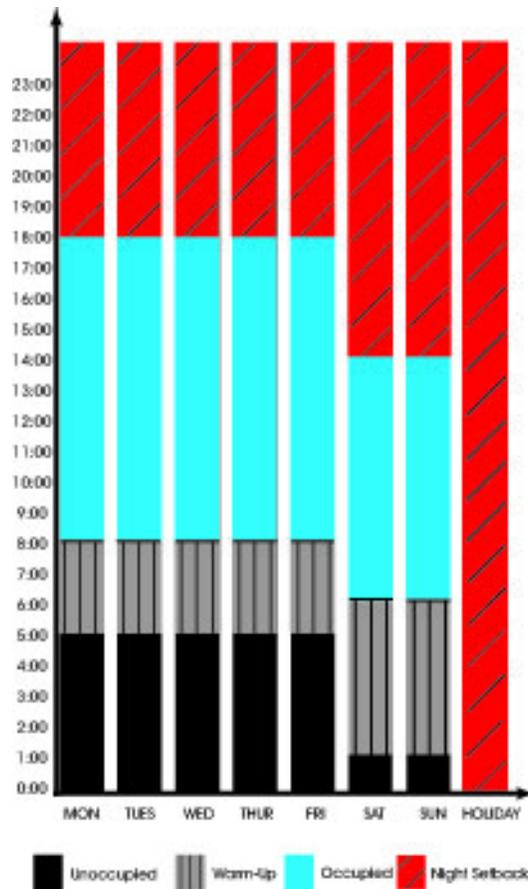


Figure 8-6: Example of schedule modes

To configure the schedule to work this way would require three schedules. The first schedule would control the weekdays and would have **AD** set for monday through friday. To set the times for the various schedule states, **OC** would be set to 8:00, **UN** to 18:00, **NS** to 0:00, and **WO** to 5:00. The second schedule would cover the weekends and would have **AD** set for saturday and sunday. For this schedule **OC** would be set to 6:00, **UN** to 14:00, **NS** to 0:00, and **WO** to 1:00. Finally, the third schedule would have **AD** set for holidays and **AO** set to "1=Unoccupied".

When the GPC is configured to receive schedules from a host controller, holiday schedules refer to the host system which defines the holidays in each month of the year. GPC also has the ability to define holidays in F900; **HO**-F900; **H9**. If the Schedule's (**AD**) **Active Days** attribute has bit 7 set for a currently active schedule, then the controller will follow that schedule when the holiday bit is sent from the host or if the current day corresponds to one of the holidays set in F900; (**H0**) through (**H9**). If the host broadcasts a holiday, and the GPC does not have a schedule with a holiday schedule, the (**FB**) **Schedule Feedback** will reflect an Inactive Holiday Today when this occurs until the Holiday Flag is cleared.

### 8.6.3 FEEDBACK ITEMS

Each Local Schedule includes several feedback properties, providing users with a way to determine past, present, and future schedule activity. These properties include **(AY) Schedule Was Active Yesterday**, **(AN) Schedule is Active Now**, **(AT) Schedule Will Be Active Tomorrow**, **(FB) Schedule Feedback**, **(CU) Currently Unoccupied Flag**, **(CW) Currently In Warmup Flag**, **(CO) Currently Occupied Flag**, **(CS) Currently in Setback Flag**, **(SO) Currently On Flag** and **(SF) Currently Off Flag**.

The **(FB) Schedule Feedback** attribute provides further clarification as to why the current schedule is in the state it is in. These variables include "Inactive Today", "Inactive Holiday Today", "All Day Override", "Value at Midnight" and "Tracking Time Entry".

## 8.7 HOLIDAYS

The Holidays area provides a method of defining up to 10 days which could be considered holidays, thereby allowing Local Schedules to operate in a specific manner. Attributes **(H0) Programmed Holiday** through **(H9) Programmed Holiday** permit users to define Month, Date, and Year information for a holiday.

System	Inputs	Outputs	STATbus	Programs	Control Loops	Scheduling	Da
Schedule Summary   Main Schedule   Host Schedule   Broadcast Schedule   Occupancy Extension   Local Schedules   <b>Holidays</b>							
F900	DH	✓	Today's Holiday Status	<input checked="" type="radio"/> No	<input type="radio"/> Yes		
F900	TH	✓	Tomorrow's Holiday Status	<input checked="" type="radio"/> No	<input type="radio"/> Yes		
F900	H0	✓	Programmed Holiday			<input type="text" value="0/0/00"/>	
F900	H1	✓	Programmed Holiday			<input type="text" value="0/0/00"/>	
F900	H2	✓	Programmed Holiday			<input type="text" value="0/0/00"/>	
F900	H3	✓	Programmed Holiday			<input type="text" value="0/0/00"/>	
F900	H4	✓	Programmed Holiday			<input type="text" value="0/0/00"/>	
F900	H5	✓	Programmed Holiday			<input type="text" value="0/0/00"/>	
F900	H6	✓	Programmed Holiday			<input type="text" value="0/0/00"/>	
F900	H7	✓	Programmed Holiday			<input type="text" value="0/0/00"/>	
F900	H8	✓	Programmed Holiday			<input type="text" value="0/0/00"/>	
F900	H9	✓	Programmed Holiday			<input type="text" value="0/0/00"/>	

Figure 8-7 : Holidays

### 8.7.1 HOLIDAY FEEDBACK

The Holidays section provides two feedback attributes dictating present holiday status via **(DH) Today's Holiday Status**, and tomorrow's holiday status via **(TH) Tomorrow's Holiday Status**.

---

## SECTION 9: DATA MANIPULATION

*This section reviews the Data Manipulation group of objects within the GPC, which are logic-based blocks used to assist in the setup and creation of control applications.*

### IN THIS SECTION

Data Manipulation Overview .....	9-3
Programming Concepts and Techniques .....	9-3
Math .....	9-4
Logic .....	9-5
Min/Max/Avg .....	9-6
Enthalpy .....	9-7
Scale .....	9-8
Input Select .....	9-9
Staging .....	9-10
Basic Configuration .....	9-10
Staging Modes .....	9-11
Stage Interlocking.....	9-11
Timers .....	9-13



## 9.1 DATA MANIPULATION OVERVIEW

The Data Manipulation object library provides many helpful logic blocks, which can be used to setup linking logic to perform control routines. The Data Manipulation library provides multiple quantities of each object type, including:

- . **Math:** used to perform simple math functions such as Add, Subtract, etc.
- . **Logic:** used to perform boolean logic functions such as AND, OR, NOT, etc.
- . **Min/Max/Avg:** used to determine minimum, maximum, and average values.
- . **Enthalpy:** used to calculate enthalpy by referencing temperature and humidity values.
- . **Scaling:** used to create a linear, interpolation scale between defined ranges.
- . **Input Selects:** used to select one of two input values based on boolean selection criteria.

Through using any of the logic blocks listed above, you can reference the present-value in any logic object within the GPC.

### 9.1.1 PROGRAMMING CONCEPTS AND TECHNIQUES

Data Manipulation channels exist in the GPC to reduce the amount of line-by-line SPL programming that one would need to write to carry out advanced control functions. Each GPC hardware model provides several of each object type to significantly reduce or eliminate the need to write SPL logic. The following are some helpful concepts and techniques to keep in mind when using these channels.

#### 9.1.1.1 MAKE THE (ON) CHANNEL NAME UNIQUE

The GPC supports the ability to allow each channel's name to be assigned a custom value. By default, the software uses generic names for channels. For ease of programming and flow, it is strongly recommended that you change the **(ON) Channel Name** attribute of any used Data Manipulation channel. This allows you not only to keep track of which channels have been used, but also allows you to easily troubleshoot your linked logic.

#### 9.1.1.2 THE (CV) CURRENT VALUE ATTRIBUTE

The **(CV) Current Value** attribute of each block (with the exception of Min/Max/Average blocks) is always the result of the logic operation. The value can be referenced by other object functions of the GPC or shared amongst one another if desired.

## 9.2 MATH

Math channel are used to apply a chosen math operator against two defined channel attributes. Configuration involves referencing the two channel attributes, as well as selecting the operators. The object's **(CV) Current Value** will reflect the result of the math operation.

The **(OP) Operation** attribute specifies the math operator applied against the first and second term. The choices for operation that can be used are displayed in Table 9-1

*Table 9-1: Math Object Operator Choices*

Value	Operation
0	Disabled
1	Addition
2	Subtraction
3	Multiplication
4	Division
5	Minimum
6	Maximum
7	Average
8	Square Root (Input 1)

Each Math channel also provides boolean logic feedback on mapped Input values through attributes **(V1) Input 1's Value**, and **(V2) Input 2's Value**. These values can be monitored from the Math channel without needing to continuously navigate between referenced variables and the Math channel.

In addition to the available math operators, each Math Channel provides boolean logic feedback on math values. Through several additional attributes, users can obtain feedback on Input 1 versus Input 2. Boolean logic provided includes:

- . Greater Than - using the **(GT) Input 1 is > Input 2** attribute
- . Greater Than Equal - using the **(GE) Input 1 is >= Input 2** attribute
- . Less Than - using the **(LT) Input 1 < Input 2** attribute
- . Less Than Equal - using the **(LE) Input 1 is <= Input 2** attribute
- . Equals - using the **(ET) Input 1 is = Input 2** attribute

### 9.3 LOGIC

Logic objects are used to perform logical operations using selectable channel attributes and a choice of operator. Up to eight (8) referenced channel attributes can be referenced in a Logic channel. The channel's **(CV) Current Value** will reflect that result of the logic operation.

The **(OP) Operation** attribute specifies the logic operator applied against the referenced channel attributes. The choices for operator that can be used are displayed in Table 9-2.

Table 9-2: Logic Object Operator Choices

Value	Operation	Notes
0	Disabled	Disables the object.
1	OR	Performs a logical "OR" on all of the referenced channel attributes. If any of the referenced channel attributes are true (value of 1), <b>(CV) Current Value</b> = true. If all of the referenced channel attributes are false (value of 0), then <b>(CV) Current Value</b> = false.
2	AND	Performs a logical "AND" on all of the referenced channel attributes. If all of the referenced channel attributes are true (value of 1), <b>(CV) Current Value</b> = true. If any of the inputs are false (value of 0), then <b>(CV) Current Value</b> = false.
3	NOT	Performs a logical "NOT" against the first referenced channel attribute (I1 and A1).
4	XOR	Performs a local "XOR" on all of the referenced channel attributes. If any one of the referenced channel attributes are true (value of 1), then <b>(CV) Current Value</b> = 1. Otherwise, <b>(CV) Current Value</b> = false.
5	NAND	Performs a logical "NAND" on the referenced channels attributes. This combines an AND in series with a NOT. If both inputs are true the result will be false. If both inputs are false the result will be true.
6	NOR	Performs a logical "NOR" on the referenced channel attributes. This combines an OR in series with a NOT. If both inputs are False, the result will be True.
7	XNOR	Performs a logical "XNOR" on the referenced channel attributes. This is an XOR that inverts the output. If both inputs are False, the result will be True, likewise if both inputs are True, the result will be True.

Feedback values relative to the status of each referenced channel attribute are provided in attributes **(V1) Input 1's Value** through **(V8) Input 8's Value**.

### 9.4 MIN/MAX/AVG

Min/Max/Avg objects are used to calculate the minimum, maximum, and average values of reference channel attributes within the block. Up to four (4) referenced channel attributes can be referenced by each Min/Max/Avg block.

The **(HV) High Value** attribute will output the highest value of all referenced channel attributes.

The **(LV) Low Value** attribute will output the lowest value of all referenced channel attributes.

The **(AV) Average Value** attribute will reference the arithmetic mean of all referenced channel attributes.

To enable calculations, set **(CE) Enable Calculations** = 1 (Yes)

Input Select		Min Max Average	Math	Logic	Scale	Staging	Enthalpy			
1	2	3	4	5	6	7	8	9	10	11
F301 ON	✓	Average Channel Name	Min/Max/Avg 1							
F301 HV	✓	High Value	72.0							
F301 AV	✓	Average Value	36.0							
F301 LV	✓	Low Value	0.0							
F301 DT	✓	PUP Datatype for Values	253							
F301 I1	✓	Input Channel 1	FE01							
F301 A1	✓	Input Attribute 1	CV							
F301 I2	✓	Input Channel 2	FE02							
F301 A2	✓	Input Attribute 2	CV							
F301 I3	✓	Input Channel 3								
F301 A3	✓	Input Attribute 3								
F301 I4	✓	Input Channel 4								
F301 A4	✓	Input Attribute 4								
F301 CE	✓	Enable Calculations?	<input type="radio"/> No <input checked="" type="radio"/> Yes							

Figure 9-1 Min Max Average

## 9.5 ENTHALPY

Enthalpy objects are used to calculate enthalpy based on a referenced temperature and referenced relative humidity value. The referenced values can be connected to inputs, retrieved using Remap Objects, SPL attributes, etc.

The **(CV) Current Value** attribute is the result of this calculation.

The screenshot shows a configuration window for an Enthalpy object. At the top, there are tabs for 'Input Select', 'Min Max Average', 'Math', 'Logic', 'Scale', 'Staging', and 'Enthalpy'. Below the tabs are four numbered buttons (1, 2, 3, 4), with button 1 selected. The main area contains a list of attributes for object F501, each with a green checkmark and a corresponding input field:

F501 ON	✓ Enthalpy Name	Enthalpy 1
F501 DT	✓ PUP Datatype	248
F501 TV	✓ Dry Bulb Temperature Value	72.000
F501 HV	✓ % Relative Humidity Value	55.700
F501 CV	✓ Current Value	27.612 English = Btu's per lb of dry air; Metric = Kilojoules per kg of dry a
F501 TC	✓ Temperature Input Channel	FE02
F501 TA	✓ Temperature Input Attribute	CV
F501 HC	✓ Humidity Input Channel	FE03
F501 HA	✓ Humidity Input Attribute	CV

Figure 9-2 - Enthalpy Channel Example

## 9.6 SCALE

Scale objects perform a linear interpolation between two known points, which can be used to scale a single value within programming by looking up values that lie along the created linear segment. A referenced channel attribute's value is applied against the scale. As a result, **(CV) Current Value** will indicate the calculated scale value.

Attributes **(X1) Input Range X1 Value**, **(X2) Input Range X2 Value** and **(Y1) Output Range Y1**, **(Y2) Output Range Y2** indicate the x- and y-coordinate values to be used for the starting and ending points of the line segment. Both x-coordinate and y-coordinate values must be given in engineering units of your referenced channel attribute input.

Attribute	Status	Value
F711 ON	✓	Scale Name
F711 CV	✓	Current Value
F711 DT	✓	PUP Datatype for Values
F711 IC	✓	Input Channel
F711 IA	✓	Input Attribute
F711 X1	✓	Input range X1 value
F711 X2	✓	Input range X2 value
F711 Y1	✓	Output range Y1 value
F711 Y2	✓	Output range Y2 value
F711 IV	✓	Input's Current Value

Figure 9-3 - Scaling Example

In the example shown above, we have configured an Input range for 0 and 100, and our output range for 100 to 200. In this example, when our input value (which is linked to Analog Output 1; present-value) is at a value of 10.0, the result of the scale will output a value of 110.0.

## 9.7 INPUT SELECT

Input Select objects allow you to choose one of two referenced channel attribute values based on a true/false input status. To use an Input Select object, you must reference two channel attributes, along with a selection criteria reference.

If the selection criteria is false (value of 0), **(CV) Current Value** will equal the value of the first channel attribute reference (I1, A1). If the selection criteria is true (value other than 0), then **(CV) Current Value** will equal the value of the second channel attribute reference (I2, A2).

1	2	3	4	5	6	7	8	9	10	11	12	13	14
F011 ON	✓	Input Select Name	Input Select 1										
F011 CV	✓	Current Value	72.0										
F011 DT	✓	PUP Datatype for Values	253										
F011 I1	✓	Input Channel 1	FE01										
F011 A1	✓	Input Attribute 1	CV										
F011 I2	✓	Input Channel 2	FE02										
F011 A2	✓	Input Attribute 2	CV										
F011 SC	✓	Select Channel	F601										
F011 SA	✓	Select Attribute	CV										
F011 V1	✓	Input 1's Value	0.0										
F011 V2	✓	Input 2's Value	72.0										
F011 SV	✓	Select Input's Value	<input type="radio"/> 0 <input checked="" type="radio"/> 1										
F011 FB	✓	Feedback	<input checked="" type="radio"/> Working Properly <input type="radio"/> Unable To Read Inputs										

Figure 9-4 - Input Select Example

## 9.8 STAGING

Staging objects are used to perform staging of binary outputs for control related purposes, but can also be used to trigger other control logic if deemed necessary. Each Staging object provides support for multiple staged outputs (as few as two, as many as eight), each with a dedicated setpoint, feedback status, and runtime timer. Stages can be transitioned for lead/lag, and wear leveling.

### 9.8.1 BASIC CONFIGURATION

The following basic configuration items should be taken into account prior to defining the staging mode.

#### 9.8.1.1 CONFIGURING THE INPUT

The Staging object will energize and de-energize stages based on the value received from the configured input. The input reference is configured using attributes **(IC) Input Channel** and **(IA) Input Attribute**.

When the Staging object is activated, the current value of the input will be reflected in attribute **(IV) Input Value**.

#### 9.8.1.2 CONFIGURING THE NUMBER OF STAGES

Determine how many output stages you wish to have. This is defined using the **(NS) Number of Stages <Max Loading>** attribute. Each Staging object can support as few as two outputs, or as many as eight output stages.

When you have successfully configured the entire Staging object for control, the control value for each Stage is defined in attributes **(S1) Stage 1 Status** through **(S8) Stage 8 Status**. In Digital Output control methods, the corresponding stage status attribute would be configured through a remap channel to the appropriate digital output.

#### 9.8.1.3 CONFIGURING THE LEAD/LAG/LEVELING MODE

Output stages can be enabled/disabled based on Normal Mode, or through Wear Leveling.

In the Normal Mode, stage outputs will energize in a classic First On / Last Off method. For example, when stages are called, they will be energized in logical order (Stage 1, Stage 2, Stage 3,...). When de-energizing occurs, the last stage on will be turned off first (Stage 8, Stage 7, Stage 6,...).

In Wear Leveling Mode, stages will be turned on and off based on the runtimes for each stage. This method allows each mechanical stage to be used for an even amount of time - thereby increasing the lifespan of equipment. Runtimes are tracked for each stage through attributes **(R1) Stage 1 Runtime** through **(R8) Stage 8 Runtime**. When stages are called, stages are energized based on the least amount of runtime.

#### 9.8.1.4 CONFIGURING THE CONTROL SIGN

Staging is commonly used for heating or cooling. This is controlled by how setpoints control the stages. Attribute **(IS) Invert the Setpoints? <Higher Stages = Lower Numbers>** essentially commands how staging will work.

When set to *False*, the Staging object will work in a Heating-like mode, where stage outputs are enabled as the input variable exceeds defined setpoint(s).

When set to *True*, the Staging object will work in a Cooling-like mode, where stage outputs are enabled as the input variable falls below defined setpoint(s).

## 9.8.2 STAGING MODES

The Staging object provides two different modes of how outputs can be staged. The mode is directly controlled through **(SM) Staging Mode**. There are two options that can be selected for staging.

### 9.8.2.1 DELAY ON/DELAY OFF

The Delay On/Delay Off mode utilizes two definable setpoints to enable and disable stages based on timed intervals. When Delay On/Delay Off mode has been selected, the **(P1) Setpoint 1** and **(P2) Setpoint 2** are used as the unloading and loading setpoints where **(P1)** is the unload point and **(P2)** is the load point. These setpoints determine when stages will be disabled (unloaded) and enabled (loaded).

When the referenced input exceeds the value defined in **(P2) Setpoint 2**, output stages will be energized (turned on) based on attribute **(LD) Loading Interval <Seconds>**. In this scenario, the first stage will be energized, and wait the amount of time specified in **(LD) Loading Interval <Seconds>**. Once the time has expired, the next stage will be energized, followed by the delay specified in **(LD) Loading Interval <Seconds>**. This process will repeat until all stages have been energized.

When the referenced input falls below the value defined in **(P1) Setpoint 1**, output stages will be de-energized (turned off) based on attribute **(UD) Unloading Interval <Seconds>**. In this scenario, the last stage energized during loading will be de-energized, and wait the amount of time specified in **(UD) Unloading Interval <Seconds>**. On the time has expired, the next stage will be de-energized, followed by the delay specified in **(UD) Unloading Interval <Seconds>**. This process will repeat until all stages have been de-energized.

For troubleshooting and convenience, two timers are provide to allow users to know when the next loading or unloading event will occur. These properties, **(LR) Seconds Until Next Loading Event Could Occur** and **(UR) Seconds Until Next Unloading Event Could Occur**, can be found in the Staging object and monitored using an engineering tool or front-end.

### 9.8.2.2 THRESHOLD BASED STAGING

The Threshold Based Staging mode utilizes a setpoint for each individual stage, rather than a single setpoint. When Threshold Based Staging mode has been selected, the object transition its properties to provide up to eight setpoints properties, defined as **(P1) Stage 1 Setpoint** through **(P8) Stage 8 Setpoint**.

Stages will be energized (turned on) when the referenced input has exceeded each defined stage setpoints. In the event that a large value increase occurs, multiple stages will be energized in a time delayed manner based on the configuration of **(LD) Loading Interval <Seconds>**.

When the referenced input falls below each defined setpoint, output stages will be de-energized (turned off). In the event of a large decrease of the input value, multiple stages will be de-energized in time delayed manner based on the configuration of **(UD) Unloading Interval <Seconds>**.

## 9.8.3 STAGE INTERLOCKING

Staging can also be subject to interlocking. Interlocking may be used to lock out stages in certain situations (e.g. supply air temperature or outside air temperature exceeds a specific setpoint value).

The Interlock input is defined using attributes **(CI) Interlock Channel** and **(AI) Interlock Attribute**. When the Interlock input is a non-zero value, all of the Stages will become interlocked, whereas a zero value will disable interlocking - allowing the Staging object to resume normal operations.

The state of each stage (enabled/disabled) when the Interlock input is a non-zero value is controlled through attribute **(LP) Interlock Map**. When a specific stage has a check mark next to it, this commands the Interlock routine to energize (turn on) the corresponding stage. When a specific stage has no check

mark next to it, this commands the Interlock routine to de-energize (turn off) the corresponding stage. For troubleshooting purposes, attribute **(IF) Interlock Feedback** will provide diagnostic feedback as to the current status of the Interlocking process.

## 9.9 TIMERS

The Timers channel attributes can be used as timers for counting upwards or downwards. A maximum of ten (10) counter attribute sets are provided within the Timers channel. While the Timers channel does not provide any direct control or manipulation of data, it can be used with SPL or other built-in logic blocks to perform time-based actions.

### 9.9.0.1 CONFIGURATION

Each counter set provides users with the ability to configure a mode for counting (e.g. **M0**). The following table describes each mode available for configuration.

Table 9-3 Timer Modes

Mode (M0)	Notes
Simple Downcounter <no clearing -w- 0>	Provides count-down functionality. To use, the counter property (e.g. C0) must first be initialized with the value you wish to start from. This can be done programmatically via SPL or through manual writes.  Any write attempted to the counter while counting down will be ignored, thereby not resetting the count down.
Simple Downcounter <clearing with 0 allowed>	Provides count-down functionality with the ability to zero the counter.
Triggered Downcounter <any positive # restarts counter>	Provides count-down functionality with the ability to perform a counter restart if a positive numeric value is written to the corresponding Timer' counter attribute (e.g. C0). The counter will then be reset to the value defined in Reset Value attribute (e.g. V0).
Triggered Downcounter <any positive # restarts & 0 clears>	Provides count-down functionality with the ability to perform a counter restart if a positive numeric value is written to the corresponding Timer' counter attribute (e.g. C0). The counter will then be reset to the value defined in Reset Value attribute (e.g. V0).  In addition, a counter can be zero valued, thereby disabling the counter until the next time a positive value has been written.
Simple Upcounter	Provides count-up functionality. When configured, the timer will count upward forever.
Simple Clearable Upcounter	Provides count-up functionality with the ability to clear the counter using a zero value.



---

# SECTION 10: DATA MOVEMENT AND STORAGE

*This section reviews the Data Movement group of objects within the GPC, which are logic-based blocks which can be used to move values from one location to another within the controller, broadcast values to a group of devices, as well as store generic values.*

## IN THIS SECTION

Data Movement Overview .....	10-3
Broadcasting Concepts .....	10-4
Broadcasts .....	10-4
Broadcasting Concepts .....	10-4
Sending a Broadcast.....	10-4
Receiving a Broadcast .....	10-4
Feedback and Status Information .....	10-4
Persisted/Remap Values .....	10-5
Configure the Operating Mode.....	10-5
Defining Inputs and Outputs.....	10-5
Configuring the Triggering Mode and Input.....	10-5
Data Coercion Mode .....	10-5
Feedback Status .....	10-5
Alarm Setup .....	10-6



## 10.1 DATA MOVEMENT OVERVIEW

The Data Movement object library provides many helpful logic blocks, used to move data from one part of the controller to another.

- . **Broadcasts:** used to send/receive information to multiple controllers without writing complex logic.
- . **Persisted/Remap Values:** used to move data back and forth between two different points local to the GPC, as well as store generic values.
- . **Alarm Setup:** enables alarm broadcasts.

### 10.1.1 PROGRAMMING CONCEPTS AND TECHNIQUES

Data Movement objects exist in the GPC to reduce the amount of line-by-line SPL programming that one would need to write to carry out advanced control functions. Each GPC hardware model provides several of each object type to significantly reduce or eliminate the need to write SPL logic. The following are some helpful concepts and techniques to keep in mind when using these objects.

#### 10.1.1.1 MAKE THE (ON) CHANNEL NAME UNIQUE

The GPC supports the ability to allow each channel's name to be assigned a custom value. By default, the software uses generic names for channels. For ease of programming and flow, it is strongly recommended that you change the **(ON) Channel Name** of any used Data Movement channel. This allows you not only to keep track of which channels have been used, but also allows you to easily troubleshoot your linked logic.

## 10.2 BROADCASTS

Broadcast objects allow GPC to send and receive values over the network at configured time intervals. Each GPC controller support up to a maximum of eight (8) broadcast objects. When an object is configured to send or receive a broadcast, **(CV) Current Value** will display the value being sent or received from the network.

### 10.2.1 BROADCASTING CONCEPTS

Before configuring broadcasts on your BACnet network, there are a few concepts that should be followed when performing Broadcasts.

#### 10.2.1.1 OUTSIDE AIR TEMPERATURE BROADCASTS

If you intend on sending/receiving an Outside Air Temperature broadcast to older PUP-based devices, such as DX1, you must send/receive the broadcast using F000 (Broadcast 0).

#### 10.2.1.2 SCHEDULE BROADCASTS

If you intend on sending/receiving a Schedule broadcast to older PUP-based devices, such as DX1, you must send/receive the broadcast using F005 (Broadcast 5).

### 10.2.2 SENDING A BROADCAST

To configure an object to send a broadcast, perform the following steps:

1. Configure **(BM) Broadcast Mode** = Send (1).
1. Reference the channel attribute you wish to broadcast by configuring **(IC) Input Channel** and **(IA) Input Attribute**.
2. Configure **(BZ) Broadcast Zone/Global** accordingly.
3. Configure **(ZN) Zone Number** for the controller zone you wish to send the broadcast to.
4. Configure **(BT) Minutes Between Broadcasts**. Determine the time interval, in minutes, you wish to have the GPC send the broadcast.

### 10.2.3 RECEIVING A BROADCAST

To configure an object to receive a broadcast, perform the following steps:

1. Configure **(BM) Broadcast Mode** = Receive (2).
2. Configure **(ZN) Zone Number** for the controller zone you wish to receive from.
3. Configure **(BZ) Broadcast Type** accordingly for Zone or Global Broadcast.

### 10.2.4 FEEDBACK AND STATUS INFORMATION

Broadcast objects provide a feedback attribute, **(FB) Feedback Status**, that provides up to date information regarding the health of the object. This attribute can be monitored via SoloPro or even an operator workstation or web server. The following table provides a list of the messages and statuses.

## 10.3 PERSISTED/REMAP VALUES

Persisted/Remap Values provide two functions:

- provide a means to move information from one place to another without having to write line-by-line SPL programming.
- provide a storage area for programmatic values, such as setpoints and other data
- 

A maximum of sixty-four (64) Persisted/Remap Value objects are supported by GPC controllers.

### 10.3.1 CONFIGURE THE OPERATING MODE

To use this object, you must first decide how the object will operate through attribute **(OM) Operating Mode**. Persisted/Remap Values may operate as:

- **Static Values** - functioning as a register value or storage holder.
- **Pull Data In and Push Data Out** - functioning in a manner where a referenced input is written to a referenced output.
- **Pull Data In** - functioning in a manner where data is pulled in and stored to **(CV) Current Value**.
- **Push Data Out** - functioning in a manner where data written to **(CV) Current Value** is pushed to a referenced output.

### 10.3.2 DEFINING INPUTS AND OUTPUTS

Inputs channel attributes (if pulling and pushing data) must be defined. Each Persisted/Remap Value contains two (2) Inputs. A single input can be defined for pulling data through attributes **(C1) Input Channel 1** and **(A1) Input Attribute 1**. However, an additional input can be defined for read/write trigger fail scenarios on specific inputs via **(C2) Input Channel 2** and **(A2) Input Attribute 2**.

A single output channel attribute, defined through attributes **(OC) Output Channel** and **(OA) Output Attribute**, defines where data will be written to for Push Data operations.

### 10.3.3 CONFIGURING THE TRIGGERING MODE AND INPUT

Attribute **(TM) Triggering Mode** defines how data is read/written. By default, No Trigger (1) reads and writes data upon any change in input value. However, reads and writes can be triggered based on a defined trigger input, defined via **(TC) Trigger Channel** and **(TA) Trigger Attribute**. When defined, values can be read and written only when triggered, or a specific input value should be read/written for programmatic purposes.

### 10.3.4 DATA COERCION MODE

Attribute **(CM) Datatype Coercion Mode** provides the ability to either automatically coerce data into a specific PUP Datatype, or receive it based on the native datatype read from the corresponding Input. By default, data is automatically coerced based on the value read by the input.

### 10.3.5 FEEDBACK STATUS

Attribute **(FB) Feedback Status** provides generalized details regarding health of the object. Information available from this attribute includes either good health, or specific configuration errors that may be present.

### 10.4 ALARM SETUP

The Alarm Setup page defines whether or not alarms generated by the GPC will be broadcasted onto the PUP network. Attribute **(BA) Broadcast Alarms?** controls whether or not broadcasts are sent. In order to broadcast, the GPC must receive the network token or be a full administrator.

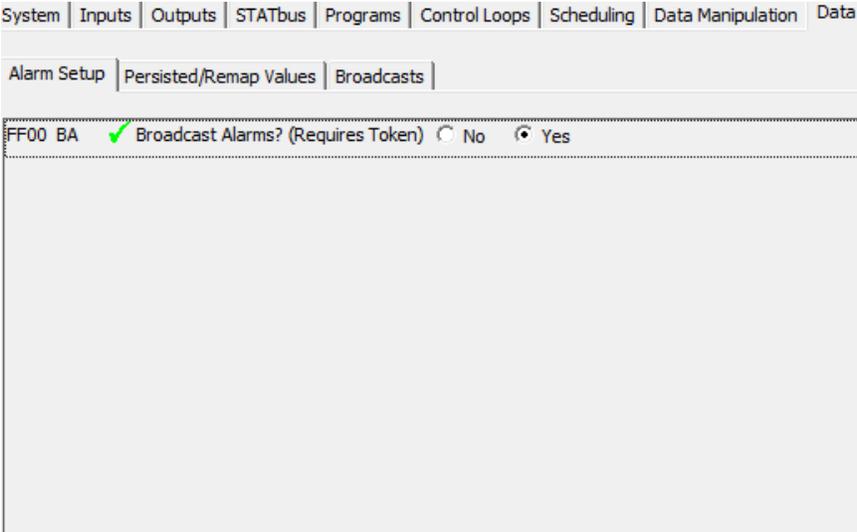


Figure 10-1 : Alarm Setup

---

# SECTION 11: SPECIAL MODES

*This section describes control objects available under the Special Modes category of the GPC platform.*

## IN THIS SECTION

Summary of Special Modes .....	11-3
Fire Mode .....	11-4
Comm Failure .....	11-5



## 11.1 SUMMARY OF SPECIAL MODES

The Summary section provides bit indications as to whether or not the GPC is in Fire Mode or Communication Failure mode. These bits are available for custom programming purposes.

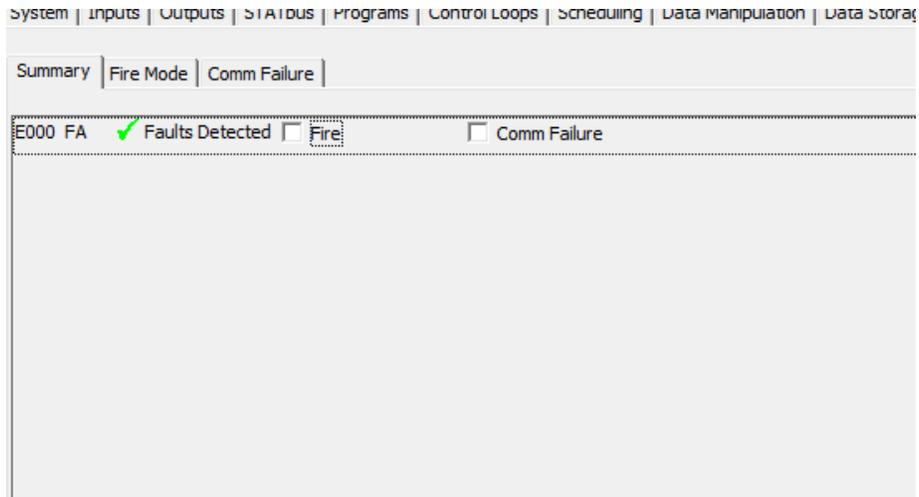


Figure 11-1 : Summary

## 11.2 FIRE MODE

The Fire Mode section provides the ability to configure certain aspects of Fire Mode, including broadcasting, timers, and a methodology of clearing fire mode. Attribute **(CV) Current Value** indicates whether or not the controller is currently in Fire Mode.

Data Manipulation		Data Storage And Movement		Special Modes		[Debug]	
<div style="display: flex; border-bottom: 1px solid black;"> <span style="margin-right: 10px;">Summary</span> <span style="margin-right: 10px; border: 1px solid black; padding: 2px;">Fire Mode</span> <span style="border: 1px solid black; padding: 2px;">Comm Failure</span> </div>							
E001	CV	✓	Current value	<input checked="" type="radio"/>	Normal	<input type="radio"/>	In Fire Mode
E001	BF	✓	Broadcast Fire Mode? (When Entering Only)	<input checked="" type="radio"/>	No	<input type="radio"/>	Yes
E001	TF	✓	Time to Remain in Fire Mode		<input type="text" value="45"/>		Minutes
E001	FM	✓	Time Remaining in Fire Mode		<input type="text" value="0"/>		Seconds
E001	FB	✓	Fire Mode Feedback	<input checked="" type="radio"/>	Not In Fire Mode	<input type="radio"/>	Locked in By Local Input
				<input type="radio"/>	Waiting for Timer to Expire	<input type="radio"/>	Waiting To Be Manually Cleared
E001	MC	✓	Manually Clear Fire Mode?	<input checked="" type="radio"/>	Do Nothing	<input type="radio"/>	Clear It

Figure 11-2 : Fire Mode

### 11.3 COMM FAILURE

The Comm Failure section of GPC is used to configure aspects of Communication Failure, as well as obtain diagnostic information for programmatic purposes.

The **(CV) Current Value** indicates whether there is a comm fail or if it has a good status. The **(OS) Out of Service** attribute can be used to override the current value and simulate comm failures.

Attributes **(FM) Failure Mode**, **(FD) Failure Mode Delay Time**, and **(BD) Failure Mode Boot Delay** configure the mode and delay times for indicating communication failures.



---

# APPENDIX A: CHANNEL ATTRIBUTES

The following tables contain a list of Public Unitary Protocol (PUP) attribute and channel assignments for the GPC. Each attribute is given with its PUP channel assignment, PUP data type, access code, where it is stored and a brief description of its function.

## IN THIS SECTION

System Channel, FF00 .....	A-3
StatBus Channels 1-3, FF01-FF03 .....	A-7
Universal Input Summary, FE00 .....	A-8
Universal Inputs 1-24, FE01-FE18.....	A-11
Digital Input Summary, FE30 .....	A-15
Digital Inputs 1-8, FE31-FE38.....	A-17
Occupancy Extension, FC01.....	A-19
Analog Output Summary, FD00 .....	A-20
Analog Outputs 1-12, FD01-FD0C.....	A-22
Digital Output Summary, FB00.....	A-25
Digital Outputs 1-12, FB01-FB0C .....	A-28
PID Control 1-12, FA01-FA0C.....	A-31
Thermostatic Control 1-12, FA11-FA1C .....	A-34
Floating Point Control 1-2, FA21-FA22 .....	A-36
Schedule Summary, F900.....	A-39
Schedules 1-8, F901-F908 .....	A-42
Scales 1-12, F711-F71C .....	A-45
Piecewise Curves 1-8, F701-F708.....	A-46
Enthalpy 1-4, F501- F504 .....	A-50
Averages 1-15, F301-F30F .....	A-55
Logic 1-16, F321-F330.....	A-56
Program Summary, F200.....	A-58
Programs 1-8, F201-F208.....	A-60
Input Select 1-15, F011-F01F .....	A-62
Broadcast 0, F000.....	A-63
Broadcast 0, F000.....	A-63
Broadcast 2, F002.....	A-65
Broadcast 3, F003.....	A-66
Broadcast 4, F004.....	A-67
Broadcast 5, F005.....	A-68
Broadcast 6, F006.....	A-69
.....	A-69



## SYSTEM CHANNEL, FF00

Attribute	Data Type	Access	Storage	Default	Description
<b>BS</b>	FE	RW	NRAM	0	<b>Broadcast Sync Time?</b> specifies whether to broadcast time synchronization information  0=No 1=Yes
<b>BT</b>	FE	RW	NRAM	255	<b>Broadcast Time Interval</b> specifies the time (in minutes) between network broadcasts
<b>CM</b>	FE	RO	RAM	255	<b>Manufacturer</b> indicates the factory-set manufacturer number for the controller. ( <b>CM</b> for American Auto-Matrix controllers is always 255)
<b>CP</b>	FE	RW	NRAM	0	<b>Communication Speed</b> the rate at which the controller communicates.  0=9600 6=38400 7=19200 8=115.2k 9=57.6k
<b>CR</b>	FE	RW	NRAM	0	<b>Configure Remote I/O</b> specifies options for the configuration of remote I/O device located on the STATbus.  0=Normal 1=GPC to bus 2=Edit I/O GIDs
<b>CT</b>	FE	RO	RAM	105	<b>Controller Type</b> factory-set controller type number for the controller. <b>CT</b> for the SBC-GPC is 105.
<b>DA</b>	FE	RO	RAM		<b>Day of Week</b> the current day of the week  0=Monday 1=Tuesday 2=Wednesday 3=Thursday 4=Friday 5=Saturday 6=Sunday
<b>DE</b>	FE	RW	RAM	0	<b>Default Enable</b> used to return all attributes in the controller to their default values.  0=Normal operation 197=set attributes to their default values.
<b>DS</b>	FE	RO	NRAM	0	<b>Daylight Savings Status</b> specifies whether the current date is included in daylight savings time.  0=No 1=Yes
<b>DT</b>	E4	RW	RAM		<b>Current Date</b> the current date displayed as MM/DD/YYYY
<b>EM</b>	FE	RW	NRAM	0	<b>Engineering Units</b> specifies the units to be used when returning values  0=English 1=Metric

Attribute	Data Type	Access	Storage	Default	Description
ET	E6	RW	NRAM	00:00	<b>Daylight Saving End Time</b> specifies the time at which daylight saving time ends.
FT	FE	RO	RAM		<b>Firmware Type</b> indicates which firmware is installed on the controller  7=GPC1 8=GPC2 9=GPC3
FC	FE	RO	NRAM	0	<b>Flash update count</b> indicates the number of times the controller has been flashed.
ID	FE	RW	NRAM		<b>Unit Number (ID)</b> specifies the controller's identification number. The value of ID defaults to the last four digits of the unit's serial number.
ND	FE	RW	NRAM	0	<b>Daylight Saving End Day</b> specifies the day on which daylight savings time ends  0 = None 1 = First Sunday 2 = First Friday 3 = Second Saturday 4 = Third Sunday 5 = Third Saturday 6 = Last Sunday 7 = Last Thursday 8 = Last Friday
NM	FE	RW	NRAM	0	<b>Daylight Saving End Month</b> specifies the month in which daylight savings time ends  0 = None 1 = January 2 = February 3 = March, 4 = April 5 = May 6 = June 7 = July 8 = August 9 = September 10 = October 11 = November 12 = December
ON	DF	RW	NRAM	"SBC-GPC1"	<b>GPC Name</b> specifies a user definable string that is used to help identify the channel or its function.
OS	FA	RO	RAM		<b>Kernel Version</b> indicates the version number of the kernel.
PD	FE	RW	NRAM	15	<b>Power-Up Delay</b> time delay (in seconds) that must elapse after the controller is reset before it begins control and alarming functions.  5-6000=# of seconds
PS	FE	RW	NRAM	0	<b>Power-up state</b> schedule state that the controller will operate in when it is first powered up or after power is restored following a power failure.  0=unoccupied 1=warmup 2=occupied 3=night setback

Attribute	Data Type	Access	Storage	Default	Description
<b>RD</b>	FE	RW	NRAM	0	<p><b>Daylight Savings Start Day</b> specifies the day on which daylight saving time begins</p> <p>0 = None 1 = First Sunday 2 = First Friday 3 = Second Saturday 4 = Third Sunday 5 = Third Saturday 6 = Last Sunday 7 = Last Thursday 8 = Last Friday</p>
<b>RM</b>	FE	RW	NRAM	0	<p><b>Daylight Savings Start Month</b> specifies the month in which daylight savings time begins</p> <p>0 = None 1 = January 2 = February 3 = March, 4 = April 5 = May 6 = June 7 = July 8 = August 9 = September 10 = October 11 = November 12 = December</p>
<b>RS</b>	FE	RW	RAM	No	<p><b>Reset the Controller?</b> used to reset the controller. Setting <b>RS</b> to 1 resets the controller.</p> <p>0=No 1=Yes</p>
<b>SN</b>	FE	RO	NRAM		<p><b>Serial Number</b> the factory-set serial number.</p>
<b>SR</b>	FE	RO	RAM		<p><b>Flash Release Code</b> the release code of the firmware currently flashed on the controller, used primarily for technical support purposes.</p>
<b>ST</b>	E6	RW	NRAM	00:00	<p><b>Daylight Saving Start Time</b> specifies the time at which daylight savings time begins.</p>
<b>TM</b>	E6	RW	RAM		<p><b>Current Time</b> indicates the current time in the form HH:MM.</p>
<b>TP</b>	FE	RW	NRAM	0	<p><b>Token Passing Type</b> defines the mode for token passing.</p> <p>0=Irresponsible Peer 1=Full Administrator</p>
<b>U1</b>	FE	RW	NRAM	65535	<p><b>Peer Unit Number</b> specifies the unit number (<b>ID</b>) for the first peer unit on the network. <b>ID</b> is set to 65,535 if there is no peer.</p>
<b>U2</b>	FE	RW	NRAM	65535	<p><b>Peer Unit Number</b> specifies the unit number (<b>ID</b>) for the second peer unit on the network. <b>ID</b> is set to 65,535 if there is no peer.</p>
<b>U3</b>	FE	RW	NRAM	65535	<p><b>Peer Unit Number</b> specifies the unit number (<b>ID</b>) for the third peer unit on the network. <b>ID</b> is set to 65,535 if there is no peer.</p>

Attribute	Data Type	Access	Storage	Default	Description
<b>U4</b>	FE	RW	NRAM	65535	<b>Peer Unit Number</b> specifies the unit number ( <b>ID</b> ) for the fourth peer unit on the network. <b>ID</b> is set to 65,535 if there is no peer.
<b>U5</b>	FE	RW	NRAM	65535	<b>Peer Unit Number</b> specifies the unit number ( <b>ID</b> ) for the fifth peer unit on the network. <b>ID</b> is set to 65,535 if there is no peer
<b>U6</b>	FE	RW	NRAM	65535	<b>Peer Unit Number</b> specifies the unit number ( <b>ID</b> ) for the sixth peer unit on the network. <b>ID</b> is set to 65,535 if there is no peer
<b>U7</b>	FE	RW	NRAM	65535	<b>Peer Unit Number</b> specifies the unit number ( <b>ID</b> ) for the seventh peer unit on the network. <b>ID</b> is set to 65,535 if there is no peer
<b>U8</b>	FE	RW	NRAM	65535	<b>Peer Unit Number</b> specifies the unit number ( <b>ID</b> ) for the eighth peer unit on the network. <b>ID</b> is set to 65,535 if there is no peer
<b>U9</b>	FE	RW	NRAM	65535	<b>Peer Unit Number</b> specifies the unit number ( <b>ID</b> ) for the ninth peer unit on the network. <b>ID</b> is set to 65,535 if there is no peer
<b>UA</b>	FE	RW	NRAM	65535	<b>Peer Unit Number</b> specifies the unit number ( <b>ID</b> ) for the tenth peer unit on the network. <b>ID</b> is set to 65,535 if there is no peer
<b>UB</b>	FE	RW	NRAM	65535	<b>Peer Unit Number</b> specifies the unit number ( <b>ID</b> ) for the eleventh peer unit on the network. <b>ID</b> is set to 65,535 if there is no peer
<b>UC</b>	FE	RW	NRAM	65535	<b>Peer Unit Number</b> specifies the unit number ( <b>ID</b> ) for the twelfth peer unit on the network. <b>ID</b> is set to 65,535 if there is no peer
<b>VE</b>	FA	RO	RAM		<b>Firmware Version</b> contains the version of the controller's firmware.
<b>VO</b>	FA	RO	RAM		<b>I/O Firmware Version</b> contains the version of the controllers I/O firmware
<b>ZN</b>	FE	RW	NRAM	0	<b>Zone Number</b> a number (from 0 to 65,535) used to group controllers so that they may be controlled together.

## STATBUS CHANNELS 1-3, FF01-FF03

Attribute	Data Type	Access	Storage	Default	Description
<b>BS</b>	FE	RO	RAM	0	<b>Bus Status</b> indicates the status of the SMARTStat Bus.  0=Ok 1=Needs configured 2=Error
<b>CD</b>	FE	RW	RAM	0	<b>Configure Device (GID)</b> used to specify a specific STATbus device that will be used with CF.
<b>CF</b>	FE	RW	RAM	0	<b>Configure Function</b> 0=Disabled 1=Blink LED
<b>G1</b>	FE	RW	NRAM	0	<b>GID Device 1</b> indicates the global identification number of device 1.
<b>G2</b>	FE	RW	NRAM	0	<b>GID Device 2</b> indicates the global identification number of device 2.
<b>G3</b>	FE	RW	NRAM	0	<b>GID Device 3</b> indicates the global identification number of device 3.
<b>G4</b>	FE	RW	NRAM	0	<b>GID Device 4</b> indicates the global identification number of device 4.
<b>G5</b>	FE	RW	NRAM	0	<b>GID Device 5</b> indicates the global identification number of device 5.
<b>G6</b>	FE	RW	NRAM	0	<b>GID Device 6</b> indicates the global identification number of device 6.
<b>G7</b>	FE	RW	NRAM	0	<b>GID Device 7</b> indicates the global identification number of device 7.
<b>G8</b>	FE	RW	NRAM	0	<b>GID Device 8</b> indicates the global identification number of device 8.
<b>G9</b>	FE	RW	NRAM	0	<b>GID Device 9</b> indicates the global identification number of device 9.
<b>GA</b>	FE	RW	NRAM	0	<b>GID Device 10</b> indicates the global identification number of device 10.
<b>GB</b>	FE	RW	NRAM	0	<b>GID Device 11</b> indicates the global identification number of device 11.
<b>GC</b>	FE	RW	NRAM	0	<b>GID Device 12</b> indicates the global identification number of device 12.
<b>GD</b>	FE	RW	NRAM	0	<b>GID Device 13</b> indicates the global identification number of device 13.
<b>ON</b>	DF	RW	NRAM	"STATbus n" where n=1 to 3	<b>StatBus Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>SM</b>	FE	RO	RAM	0	<b>Status Map</b> 0=reserved 1=unconfigured 2=duplicate

UNIVERSAL INPUT SUMMARY, FE00

Attribute	Data Type	Access	Storage	Default	Description
<b>AT</b>	FE	RW	NRAM	0	<b>Alarm Limit setup/setback time delay</b> the time (in minutes) that must elapse after transitions from occupied to unoccupied or from unoccupied to occupied before alarm limits return to their previous (pre-setup/setback) values.
<b>IP</b>	E9	RW	NRAM	0	<b>Input Polarities</b> specifies how the controller represents the input internally.  0=Normal polarity. Low Voltage is displayed as <b>CV=0</b> and High Voltage is displayed as <b>CV=1</b> . 1=Reverse polarity. Low Voltage is displayed as <b>CV=1</b> and High Voltage is displayed as <b>CV=0</b> .  <b>IP</b> is a bitmap with bit 1 corresponding to UI1, bit 2=UI2, etc. up to bit 24=UI24.
<b>LP</b>	E9	RW	NRAM	0	<b>Interlock Polarities</b> indicates the individual interlock polarity states for each UI.  0=Normal 1=Reverse  <b>LP</b> is a bitmap with bit 1 corresponding to UI1, bit 2=UI2, etc. up to bit 24=UI24.
<b>OI</b>	E9	RW	NRAM	0	<b>Overridden Inputs</b> allows the current value of the corresponding input, V1 through VO, to be overridden.  0=Normal 1=Override  <b>OI</b> is a bitmap with bit 1 corresponding to UI1, bit 2=UI2, etc. up to bit 24=UI24.
<b>ON</b>	DF	RW	NRAM	"Universal Input Summary"	<b>Universal Inputs Summary Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>PI</b>	FE	RW	NRAM	0	<b>Installer P.I.N.</b> specifies the P.I.N. code used to enter the Installer menu.
<b>PS</b>	FE	RW	NRAM	1100	<b>Service P.I.N.</b> specifies the P.I.N. code used to enter the Service menu.
<b>PU</b>	FE	RW	NRAM	0	<b>User P.I.N.</b> specifies the P.I.N. code used to enter the User menu.
<b>RE</b>	E9	RO	RAM		<b>Inputs with Unreliable Channels</b> indicates whether the reading from the corresponding input is considered reliable.  0=Reliable 1=Unreliable  <b>RE</b> is a bitmap with bit 1 corresponding to UI1, bit 2=UI2, etc. up to bit 24=UI24.
<b>V1</b>	FE	RW	RAM		<b>Current Measured Input 1 Value</b> indicates the current measured value of input 1.
<b>V2</b>	FE	RW	RAM		<b>Current Measured Input 2 Value</b> indicates the current measured value of input 2.

Attribute	Data Type	Access	Storage	Default	Description
V3	FE	RW	RAM		<b>Current Measured Input 3 Value</b> indicates the current measured value of input 3.
V4	FE	RW	RAM		<b>Current Measured Input 4 Value</b> indicates the current measured value of input 4.
V5	FE	RW	RAM		<b>Current Measured Input 5 Value</b> indicates the current measured value of input 5.
V6	FE	RW	RAM		<b>Current Measured Input 6 Value</b> indicates the current measured value of input 6.
V7	FE	RW	RAM		<b>Current Measured Input 7 Value</b> indicates the current measured value of input 7.
V8	FE	RW	RAM		<b>Current Measured Input 8 Value</b> indicates the current measured value of input 8.
V9	FE	RW	RAM		<b>Current Measured Input 9 Value</b> indicates the current measured value of input 9.
VA	FE	RW	RAM		<b>Current Measured Input 10 Value</b> indicates the current measured value of input 10.
VB	FE	RW	RAM		<b>Current Measured Input 11 Value</b> indicates the current measured value of input 11.
VC	FE	RW	RAM		<b>Current Measured Input 12 Value</b> indicates the current measured value of input 12.
VD	FE	RW	RAM		<b>Current Measured Input 13 Value</b> indicates the current measured value of input 13.
VE	FE	RW	RAM		<b>Current Measured Input 14 Value</b> indicates the current measured value of input 14.
VF	FE	RW	RAM		<b>Current Measured Input 15 Value</b> indicates the current measured value of input 15.
VG	FE	RW	RAM		<b>Current Measured Input 16 Value</b> indicates the current measured value of input 16.
VH	FE	RW	RAM		<b>Current Measured Input 17 Value</b> indicates the current measured value of input 17.
VI	FE	RW	RAM		<b>Current Measured Input 18 Value</b> indicates the current measured value of input 18.
VJ	FE	RW	RAM		<b>Current Measured Input 19 Value</b> indicates the current measured value of input 19.
VK	FE	RW	RAM		<b>Current Measured Input 20 Value</b> indicates the current measured value of input 20.
VL	FE	RW	RAM		<b>Current Measured Input 21 Value</b> indicates the current measured value of input 21.
VM	FE	RW	RAM		<b>Current Measured Input 22 Value</b> indicates the current measured value of input 22.
VN	FE	RW	RAM		<b>Current Measured Input 23 Value</b> indicates the current measured value of input 23.

---

Attribute	Data Type	Access	Storage	Default	Description
<b>VO</b>	FE	RW	RAM		<b>Current Measured Input 24 Value</b> indicates the current measured value of input 24.

## UNIVERSAL INPUTS 1-24, FE01-FE18

Attribute	Data Type	Access	Storage	Default	Description
AD	FE	RW	NRAM	0	<p><b>Alarm Delay</b> specifies the amount of time that the input must exceed a limit before an alarm condition is generated.</p> <p>0=Disable 1-255 Seconds</p>
AE	FE	RW	NRAM	0	<p><b>Alarm Enable</b> specifies the type of alarming to be used by the input.</p> <p>0=Disabled 1=Contact 0&gt;1 2=Contact 1&gt;0 3=Change of State 4=Low Limit 5=High Limit 6=Low &amp; Hi Limit 8=Fire 11=Supervise DO1 12=Supervise DO2 13=Supervise DO3 14=Supervise DO4 15=Supervise DO5 16=Supervise DO6 17=Supervise DO7 18=Supervise DO8 19=Supervise DO9 20=Supervise DO10 21=Supervise DO11 22=Supervise DO12</p>
AS	FE	RO	RAM	0	<p><b>Alarm Status</b> indicates the alarm state for the input.</p> <p>0=Normal 1=Contact 0&gt;1 2=Contact 1&gt;0 3=Change of State 5=Low Limit 6=High Limit 8=Fire 10=Runtime 11=Supervisory</p>
CV	FE	RW	RAM	0	<p><b>Current Measured Input Value</b> displays the scaled value of the associated universal input.</p>
DF	FE	RW	NRAM	0	<p><b>Display Format</b> specifies the way in which the stat will display the temperature.</p> <p>0=##d 1=##.#d 2=##dF 3=##.#dF 4=None</p>
DT	FE	RW	NRAM	254	<p><b>PUP Datatype for Input</b> specifies the PUP datatype for the input.</p>
ED	FE	RW	NRAM	60	<p><b>Extended Occupancy Duration</b> specifies the amount of time that the extended occupancy override will occur</p>
ER	FE	RW	NRMA	0	<p><b>Extended Occupancy Remaining</b> indicates the amount of time left in extended occupancy mode. This can be cleared out by writing a value of zero.</p>

Attribute	Data Type	Access	Storage	Default	Description
<b>GI</b>	FE	RW	NRAM	0	<b>GID of I/O Device</b> indicates the global identification number of the STATbus device associated with the universal input. Universal Inputs 1-12 will have a value of <b>GI</b> set equal to the serial number of the GPC. If Universal Inputs 13-24 do not have a STATbus device mapped to them, <b>GI</b> for that input will be 0.
<b>HL</b>	FE	RW	NRAM	0	<b>High Alarm Limit</b> if alarming is enabled and the current value, <b>CV</b> , rises above <b>HL</b> , then a high limit alarm will be generated.
<b>HS</b>	FE	RW	NRAM	0	<b>Alarm Limit Hysteresis</b> specifies a value that determines when the controller returns from a high- or low-alarm limit. The current value, <b>CV</b> , must be <b>HS</b> below <b>HL</b> or <b>HS</b> above <b>LL</b> before a limit return is generated.
<b>IF</b>	FC	RW	NRAM	0	<b>Input Filter Delay or Weighted Gain</b> the debounce time (in seconds) during which the input must remain stable to avoid the signal being read as a digital bounce. In the case of a bounce, the channel reliability, <b>RE</b> , is set to 1.  For analog inputs, <b>IF</b> specifies a weighted gain used to smooth the values from a fluctuating input.
<b>IP</b>	FE	RW	NRAM	0	<b>Input Polarity</b> specifies how the controller represents the input internally.  0=Normal Polarity. Low Voltage is displayed as <b>CV</b> =0 and High Voltage is displayed as <b>CV</b> =1. 1=Reverse Polarity. Low Voltage is displayed as <b>CV</b> =1 and High Voltage is displayed as <b>CV</b> =0.
<b>I#</b>	FE	RW	NRAM	255	<b>Input Index of I/O Device</b> indicates the input index of the I/O device. Onboard inputs will have the same value as the Universal Input.
<b>LL</b>	FE	RW	NRAM	0	<b>Low Alarm Limit</b> if alarming is enabled and the current value, <b>CV</b> , drops below <b>LL</b> , then a low limit alarm will be generated.
<b>MD</b>	FE	RW	NRAM		<b>Pulse Counter Mode</b> defines how the controller detects a pulse. Feature unsupported may be shown based on I/O firmware.
<b>MN</b>	FE	RW	NRAM	0	<b>Minimum Scaled Value</b> specifies the minimum scaled value in engineering units.
<b>MO</b>	FC	RW	NRAM	3.0	<b>Maximum Loop Override</b> determines the maximum value that a setpoint can be changed when association with a control loop.
<b>MX</b>	FE	RW	NRAM	0	<b>Maximum Scaled Value</b> specifies the maximum scaled value in engineering units.
<b>NP</b>	FE	RO	RAM	0	<b>Number of Pulses Accumulated</b> displays the current number of pulses detected on the digital input selected in IC.
<b>OF</b>	FB	RW	NRAM		<b>Input Offset</b> specifies an offset amount to be added to the current value.
<b>OI</b>	FE	RW	NRAM	0	<b>Override Input?</b> allows the current value of the input, <b>CV</b> , to be overridden.  0=No 1=Yes

Attribute	Data Type	Access	Storage	Default	Description
<b>ON</b>	DF	RW	NRAM	"Universal Input n" where n=1 to 24	<b>Universal Input Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>PT</b>	FA	RW	NRAM	0	<b>Pulse Threshold (For Pulse Counting Mode)</b> indicates the threshold when pulses should be counted
<b>RE</b>	FE	RO	RAM		<b>Data Reliability</b> indicates whether the reading from the input is considered reliable.  0=Reliable 1=Unreliable
<b>RH</b>	FC	RW	RAM	0	<b>Run Hours</b> indicates the number of hours <b>CV</b> =1 for the input.
<b>RL</b>	FC	RW	NRAM	0	<b>Run Limit</b> specifies a number of hours that <b>CV</b> =1 after which a run limit alarm is generated. Setting <b>RL</b> =0.0 disabled run limit alarms for the input.
<b>SC</b>	F9	R	NRAM	0	<b>Scaled Pulse Count</b> displays a scaled version of NP using the formula <b>SC</b> =NP x SF
<b>SM</b>	E9	RW	NRAM	0	<b>Schedules to Follow</b> enables scheduled alarm controlling for the associated universal input by selecting one or more of the available schedule control channels.  0=Schedule disabled 1=Schedule enabled  <b>SM</b> is a bitmap with bit #0=F901 bit #1=F902 bit #2=F903 bit #3=F904 bit #4=F905 bit #5=F906 bit #6=F907 bit #7=F908 bit #8=UI13 or SMARTStat1 bit #9=UI14SMARTStat2 bit #10=UI15SMARTStat3 bit #11=UI16SMARTStat4 bit #12=UI17SMARTStat5 bit #13=UI18SMARTStat6 bit #14=UI19SMARTStat7 bit #15=UI20SMARTStat8 bit #16=UI21SMARTStat9 bit #17=UI22SMARTStat10 bit #18=UI23SMARTStat11 bit #19=UI24SMARTStat12 bit #20=Host Schedule bit #21=Schedule Summary(F900) bit #22=Occupancy Detection(FC01)
<b>ST</b>	FE	RW	NRAM	0	<b>Sensor Type</b> specifies the type of sensor connected to the input.  0=Digital 2=MN..MX 0 to 5V 3=MN..MX 4 to 20mA 4=Curve 1 5=Curve 2 6=MN..MX 0 to 10V 7=Thermistor -30.0 to 230.0 8=MN..MX 0 to 20mA 9=SMARTStat Temperature 10=SMARTStat Humidity

Attribute	Data Type	Access	Storage	Default	Description
<b>SU</b>	FC	RW	NRAM	0	<b>Amount to Setup/Setback Alarm Limit</b> specifies a value (0.0 to 25.5) which is added to <b>HL</b> and subtracted from <b>LL</b> during scheduled unoccupied periods.
<b>TS</b>	FE	RW	NRAM	0	<b>Show the Time (STATs Only)</b> determines if the smart stat should show the current time on the display
<b>VR</b>	FE	RW	NRAM	2	<b>IVR Setting (For Pulse Counting Mode)</b> determines what the current jumper position is for the associated input.

## DIGITAL INPUT SUMMARY, FE30

Attribute	Data Type	Access	Storage	Default	Description
CV	E9	RW	RAM		<p><b>Current Digital Input Value</b> indicates the current state of the associated digital input.</p> <p>bit #0=FE31 bit #1=FE32 bit #2=FE33 bit #3=FE34 bit #4=FE35 bit #5=FE36 bit #6=FE37 bit #7=FE38</p>
IP	E9	RW	NRAM	0	<p><b>Input Polarities</b> specifies how the controller represents the input internally.</p> <p>0=Normal polarity. Low Voltage is displayed as <b>present_value=0</b> and High Voltage is displayed as <b>present_value=1</b>. 1=Reverse polarity. Low Voltage is displayed as <b>present_value=1</b> and High Voltage is displayed as <b>present_value=0</b>.</p> <p>bit #0=FE31 bit #1=FE32 bit #2=FE33 bit #3=FE34 bit #4=FE35 bit #5=FE36 bit #6=FE37 bit #7=FE38</p>
LP	E9	RW	NRAM	0	<p><b>Interlock Polarities</b> indicates the individual interlock polarity states for each Digital Input.</p> <p>0=Normal 1=Reverse</p> <p>bit #0=FE31 bit #1=FE32 bit #2=FE33 bit #3=FE34 bit #4=FE35 bit #5=FE36 bit #6=FE37 bit #7=FE38</p>
OI	E9	RW	NRAM	0	<p><b>Overridden Inputs</b> indicates any inputs which have been overridden.</p> <p>bit #0=FE31 bit #1=FE32 bit #2=FE33 bit #3=FE34 bit #4=FE35 bit #5=FE36 bit #6=FE37 bit #7=FE38</p>
ON	DF	RW	NRAM	"Digital Input Summary"	<p><b>Digital Inputs Summary Name</b> specifies a user definable string that is used to help identify the channel or its function.</p>

---

Attribute	Data Type	Access	Storage	Default	Description
RE	E9	RO	RAM		<p><b>Inputs with Unreliable Values</b> indicates whether the reading from the corresponding digital input is considered reliable.</p> <p>0=Reliable 1=Unreliable</p> <p><b>RE</b> is a bitmap with bit 1 corresponding to DI1, bit 2=DI2, etc. up to bit 4=DI4.</p>

## DIGITAL INPUTS 1-8, FE31-FE38

Attribute	Data Type	Access	Storage	Default	Description
<b>AD</b>	FE	RW	NRAM	0	<b>Alarm Delay</b> specifies the amount of time that the input must exceed a limit before an alarm condition is generated.  0=Disable 1-255 seconds
<b>AE</b>	FE	RW	NRAM	0	<b>Alarm Enable</b> specifies the type of alarming to be used by the input.  0=Disabled 1=Contact 0>1 2=Contact 1>0 3=Change Of State 8=Fire 11=Supervise DO1 12=Supervise DO2 13=Supervise DO3 14=Supervise DO4 15=Supervise DO5 16=Supervise DO6 17=Supervise DO7 18=Supervise DO8 19=Supervise DO9 20=Supervise DO10 21=Supervise DO11 22=Supervise DO12
<b>AS</b>	FE	RO	RAM	0	<b>Alarm Status</b> indicates the alarm state for the input.  0=Normal 1=Contact 0>1 2=Contact 1>0 3=Change of state 8=Fire 10=Runtime 11=Supervisory
<b>CV</b>	FE	RW	NRAM		<b>Current Input Value</b> indicates the current state of the associated digital input.
<b>DT</b>	FE	RW	NRAM	254	<b>PUP Datatype for Input</b> specifies the PUP datatype for the input.
<b>GI</b>	FE	RW	NRAM	0	<b>GID of I/O Device</b> indicates the global identification number of the STATbus device associated with the digital input. Digital Inputs 1-6 will have a value of <b>GI</b> set equal to the serial number of the GPC. If Digital inputs 7-12 do not have a STATbus device mapped to them, <b>GI</b> for that input will be 0.
<b>IF</b>	FC	RW	NRAM	0	<b>Input Filter Delay</b> the debounce time (in seconds) during which the input must remain stable to avoid the signal being read as a digital bounce. In the case of a bounce, the channel reliability, <b>RE</b> , is set to 1.
<b>IP</b>	FE	RW	NRAM	0	<b>Input Polarity</b> specifies how the controller represents the input internally.  0=Normal polarity. Low Voltage is displayed as <b>CV</b> =0 and High Voltage is displayed as <b>CV</b> =1. 1=Reverse polarity. Low Voltage is displayed as <b>CV</b> =1 and High Voltage is displayed as <b>CV</b> =0.
<b>I#</b>	FE	RW	NRAM	255	<b>Input Index of I/O Device</b> indicates the input index of the I/O device.

Attribute	Data Type	Access	Storage	Default	Description
<b>MD</b>	FE	RW	NRAM		<b>Pulse Counter Mode</b> defines how the GPC detects a pulse.  0=Rising edge 1=Falling edge
<b>NP</b>	FE	RW	NRAM		<b>Number of Pulses Accumulated</b> displays the current number of pulses detected on the digital input selected in <b>IC</b> .
<b>OI</b>	FE	RW	NRAM	0	<b>Override Input?</b> allows the current value of the input, CV, to be overridden.  0=No 1=Yes
<b>ON</b>	DF	RW	NRAM	"Digital Input n" where n=1 to 8	<b>Digital Input Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>RE</b>	FE	RO	RAM		<b>Data Reliability;</b> indicates whether the reading from the corresponding digital input is considered reliable.  0=Reliable 1=Unreliable
<b>RH</b>	FC	RW	RAM		<b>Run Hours</b> indicates the number of hours <b>CV=1</b> for the input.
<b>RL</b>	FC	RW	NRAM	0	<b>Run Limit</b> specifies a number of hours that <b>CV=1</b> after which a run limit alarm is generated. Setting <b>RL=0.0</b> disabled run limit alarms for the input.
<b>SC</b>	F9	RO	RAM	0	<b>Scaled Pulse Count</b> displays a scaled version of NP using the formula $SC=NP \times NF$
<b>SF</b>	F8	RW	NRAM		<b>Pulse Multiplier</b> specifies a scaling factor (0.000 to 65.535) that is multiplied by <b>NP</b> to obtain a scaled count.
<b>SV</b>	F8	RO	RAM		<b>Scaled Pulse Count</b> displays a scaled version of NP using the formula $SV = NP \times SF$

## OCCUPANCY EXTENSION, FC01

Attribute	Data Type	Access	Storage	Default	Description
<b>IA</b>	DF	RW	NRAM		<b>Input Attribute for Occupancy Detection</b> specifies the attribute associated with the channel specified in <b>IC</b> to be used for occupancy detection.
<b>IC</b>	DF	RW	NRAM		<b>Input Channel for Occupancy Detection</b> specifies the channel of the input to be used for occupancy detection.
<b>MD</b>	FE	RW	NRAM	0	<b>Delay Going Occupied</b> specifies the time, in seconds, that the occupancy detector status, <b>MS</b> , must be positive before the controller will switch to occupied mode.
<b>MR</b>	FE	RO	RAM		<b>Extended Occupancy Remaining</b> indicates the time, in seconds, before the controller reverts to unoccupied mode.
<b>MS</b>	FE	RO	RAM		<b>Occupancy Status</b> indicates whether the channel and attribute specified in <b>IC</b> and <b>IA</b> indicate that the monitored zone is occupied.
<b>MT</b>	FE	RW	NRAM	0	<b>Delay to Go Unoccupied</b> specifies the time, in minutes, that the controller will stay in occupied mode once occupancy is detected.
<b>ON</b>	DF	RW	NRAM	"Occupancy Detector"	<b>Occupancy Output Name</b> specifies a user definable string that is used to help identify the channel or its function.

ANALOG OUTPUT SUMMARY, FD00

Attribute	Data Type	Access	Storage	Default	Description
<b>AM</b>	E9	RW	RAM		<p><b>Outputs in Automatic Mode</b> specifies how the analog output is controlled.</p> <p>0=Manual 1=Automatic</p> <p><b>AM</b> is a bitmap with bit 1 corresponding to AO1, bit 2=AO2, etc. up to bit 12=A0C.</p>
<b>CF</b>	FE	RW	NRAM	0	<p><b>Outputs Enabled for Communication Failure</b> enables the communications failure feature on an individual basis for each analog output.</p> <p>0=Disabled 1=Enabled</p> <p>bit #0=FD01 bit #1=FD02 bit #2=FD03 bit #3=FD04 bit #4=FD05 bit #5=FD06 bit #6=FD07 bit #7=FD08 bit #8=FD09 bit #9=FD0A bit #10=FD0B bit #11=FD0C</p>
<b>ON</b>	DF	RW	NRAM	"Analog Output Summary"	<p><b>Analog Output Summary Name</b> specifies a user definable string that is used to help identify the channel or its function.</p>
<b>RE</b>	E9	RO	RAM		<p><b>Outputs with Unreliable States</b> indicates whether the reading from the corresponding analog output is considered reliable.</p> <p>0=Reliable 1=Unreliable</p> <p><b>RE</b> is a bitmap with bit 1 corresponding to AO1, bit 2=AO2, etc. up to bit 12=A0C.</p>
<b>V1</b>	FC	RW	RAM		<p><b>Current Value for Output 1</b> indicates the current value of output 1.</p>
<b>V2</b>	FC	RW	RAM		<p><b>Current Value for Output 2</b> indicates the current value of output 2.</p>
<b>V3</b>	FC	RW	RAM		<p><b>Current Value for Output 3</b> indicates the current value of output 3.</p>
<b>V4</b>	FC	RW	RAM		<p><b>Current Value for Output 4</b> indicates the current value of output 4.</p>
<b>V5</b>	FC	RW	RAM		<p><b>Current Value for Output 5</b> indicates the current value of output 5.</p>
<b>V6</b>	FC	RW	RAM		<p><b>Current Value for Output 6</b> indicates the current value of output 6.</p>
<b>V7</b>	FC	RW	RAM		<p><b>Current Value for Output 7</b> indicates the current value of output 7.</p>

---

Attribute	Data Type	Access	Storage	Default	Description
<b>V8</b>	FC	RW	RAM		<b>Current Value for Output 8</b> indicates the current value of output 8.
<b>V9</b>	FC	RW	RAM		<b>Current Value for Output 9</b> indicates the current value of output 9.
<b>VA</b>	FC	RW	RAM		<b>Current Value for Output 10</b> indicates the current value of output 10.
<b>VB</b>	FC	RW	RAM		<b>Current Value for Output 11</b> indicates the current value of output 11.
<b>VC</b>	FC	RW	RAM		<b>Current Value for Output 12</b> indicates the current value of output 12.

ANALOG OUTPUTS 1-12, FD01-FD0C

Attribute	Data Type	Access	Storage	Default	Description
<b>AM</b>	FE	RW	NRAM	0	<b>Control Mode</b> specifies how the analog output is controlled.  0=Manual 1=Automatic
<b>CB</b>	FE	RO	RAM		<b>Controlled By</b> provides feedback for how the output is currently being controlled  0=Manual write 1=Automatic 2=Fire Mode 3=Interlocked Mode 4=Communication Failure Mode
<b>CF</b>	FE	RW	NRAM	0	<b>Communication Failure Enable?</b> 0=No 1=Yes
<b>CV</b>	FC	RW	RAM		<b>Current Output Value</b> indicates the current value of the associated analog output.
<b>DT</b>	FE	RW	NRAM	252	<b>PUP Datatype for Output</b> specifies the PUP datatype for the output.
<b>FE</b>	FE	RW	NRAM	0	<b>Enable Fire Override</b> specifies if the output should be forced to a certain position when the controller is in fire mode
<b>FI</b>	FC	RW	NRAM	0	<b>Fire Position</b> specifies the failure position (0-100%) for the output to be used when a fire event is received.
<b>FP</b>	FC	RW	NRAM	0.0	<b>Communication Failure Position</b> specifies the failure position (0-100%) for the output to be used in the event of an interlock or communications failure.
<b>GI</b>	FE	RW	NRAM	0	<b>GID of I/O Device</b> indicates the global identification number of the STATbus device associated with the analog output. Analog outputs 1-6 will have a value of <b>GI</b> set equal to the serial number of the GPC. If Digital inputs 7-12 do not have a STATbus device mapped to them, <b>GI</b> for that input will be 0.
<b>HS</b>	FA	RW	NRAM	100	<b>Maximum Scaled Voltage</b> specifies the percentage of the total output for <b>CV=MX</b> .
<b>IA</b>	DF	RW	NRAM		<b>[Additional Interlock] Input Attribute</b> specifies which input attribute (if any) will be used as an interlock for the associated output.
<b>IC</b>	DF	RW	NRAM		<b>[Additional Interlock] Input Channel</b> specifies which input channel (if any) will be used as an interlock for the associated output.

Attribute	Data Type	Access	Storage	Default	Description
IL	E9	RW	NRAM	0	<p><b>Inputs for Interlocking</b> specifies which inputs (if any) are used as interlocks for the associated output (a value of 1 indicates that that input is used for interlocking).</p> <p>bit #0=FE01 bit #1=FE02 bit #2=FE03 bit #3=FE04 bit #4=FE05 bit #5=FE06 bit #6=FE07 bit #7=FE08 bit #8=FE09 bit #9=FE0A bit #10=FE0B bit #11=FE0C bit #12=FE0D bit #13=FE0E bit #14=FE0F bit #15=FE10 bit #16=FE11 bit #17=FE12 bit #18=FE13 bit #19=FE14 bit #20=FE15 bit #21=FE16 bit #22=FE17 bit #23=FE18 bit #24=FE31 bit #25=FE32 bit #26=FE33 bit #27=FE34 bit #28=FE35 bit #29=FE36 bit #30=FE37 bit #31=FE38</p>
IT	FE	RW	NRAM	0	<p><b>[Additional Interlock] Trigger</b> specifies the biasing of the trigger</p> <p>0=Trigger on zero 1=Trigger on non-zero</p>
LS	FA	RW	NRAM	0	<p><b>Minimum Scaled Voltage</b> specifies the percentage of the total output for CV=MN.</p>
MN	FE	RW	NRAM	0	<p><b>Minimum Scaled Value</b> specifies the minimum value for a scaled reading of CV, in Engineering Units.</p>
MX	FE	RW	NRAM	100	<p><b>Maximum Scaled Value</b> specifies the maximum value for a scaled reading of CV, in Engineering Units.</p>
OC	FA	RO	RAM		<p><b>Actual Output Current</b> indicates the actual current output of the associated analog output</p>
ON	DF	RW	NRAM	"Analog Output n" where n=1 to 12	<p><b>Analog Output Name</b> specifies a user definable string that is used to help identify the channel or its function.</p>
OV	FA	RO	RAM		<p><b>Actual Output Voltage</b> indicates the actual output voltage of the analog output.</p>
O#	FE	RW	NRAM	255	<p><b>Output Index of I/O Device</b> specifies the output index of the I/O device.</p>

Attribute	Data Type	Access	Storage	Default	Description
RE	FE	RO	RAM		<p><b>Data Reliability</b>  indicates whether the reading from the corresponding analog output is considered reliable.</p> <p>0=Reliable  1=Unreliable</p>
UT	FE	RW	NRAM	0	<p><b>Update Threshold</b>  specifies a threshold value by which CV must change before the output is updated.</p>

## DIGITAL OUTPUT SUMMARY, FB00

Attribute	Data Type	Access	Storage	Default	Description
AM	E9	RW	NRAM	0	<p><b>Outputs in Automatic Mode</b> specifies how the digital output is controlled.</p> <p>0=Manual 1=Automatic</p> <p>bit #0=BO1 bit #1=BO2 bit #2=BO3 bit #3=BO4 bit #4=BO5 bit #5=BO6 bit #6=BO7 bit #7=BO8 bit #8=BO9 bit #9=BO10 bit #10=BO11 bit #11=BO12</p>
CF	E9	RW	NRAM	0	<p><b>Outputs Enabled for Communication Fail</b> specifies which outputs will be enabled in the event that a communication failure is detected.</p> <p>bit #0=BO1 bit #1=BO2 bit #2=BO3 bit #3=BO4 bit #4=BO5 bit #5=BO6 bit #6=BO7 bit #7=BO8 bit #8=BO9 bit #9=BO10 bit #10=BO11 bit #11=BO12</p>
CV	E9	RW	RAM	0	<p><b>Digital Outputs which are On</b> indicates which outputs are currently on.</p> <p>bit #0=BO1 bit #1=BO2 bit #2=BO3 bit #3=BO4 bit #4=BO5 bit #5=BO6 bit #6=BO7 bit #7=BO8 bit #8=BO9 bit #9=BO10 bit #10=BO11 bit #11=BO12</p>
FE	E9	RW	NRAM	0	<p><b>Outputs Enabled for Fire</b> specifies which outputs will be enabled in the event that a fire event is detected.</p> <p>bit #0=BO1 bit #1=BO2 bit #2=BO3 bit #3=BO4 bit #4=BO5 bit #5=BO6 bit #6=BO7 bit #7=BO8 bit #8=BO9 bit #9=BO10 bit #10=BO11 bit #11=BO12</p>

Attribute	Data Type	Access	Storage	Default	Description
FI	E9	RO	NRAM	0	<p><b>Fire Positions</b> indicates which digital outputs are in fire mode.</p> <p>bit #0=BO1 bit #1=BO2 bit #2=BO3 bit #3=BO4 bit #4=BO5 bit #5=BO6 bit #6=BO7 bit #7=BO8 bit #8=BO9 bit #9=BO10 bit #10=BO11 bit #11=BO12</p>
FS	E9	RO	NRAM	0	<p><b>Communication Failure Positions</b> indicates which outputs are currently experiencing communications failures.</p> <p>bit #0=BO1 bit #1=BO2 bit #2=BO3 bit #3=BO4 bit #4=BO5 bit #5=BO6 bit #6=BO7 bit #7=BO8 bit #8=BO9 bit #9=BO10 bit #10=BO11 bit #11=BO12</p>
IM	E9	RW	NRAM		<p><b>Interlock Positions</b> indicates if the digital output is currently interlocked</p> <p>0=Normal 1=Interlocked</p> <p>bit#0=BO1 bit#2=BO2 bit#3=BO3 bit#4=BO4 bit#5=BO5 bit#6=BO6 bit#7=BO7 bit#8=BO8 bit#9=BO9 bit#10=BO10 bit#11=BO11 bit#12=BO12</p>
ON	DF	RW	NRAM	"Digital Output Summary"	<p><b>Digital Outputs Summary Name</b> specifies a user definable string that is used to help identify the channel or its function.</p>
OU	E9	RO	RAM		<p><b>Actual Output States</b> indicates the actual output states of the digital outputs.</p> <p>bit #0=BO1 bit #1=BO2 bit #2=BO3 bit #3=BO4 bit #4=BO5 bit #5=BO6 bit #6=BO7 bit #7=BO8 bit #8=BO9 bit #9=BO10 bit #10=BO11 bit #11=BO12</p>

Attribute	Data Type	Access	Storage	Default	Description
RE	FE	RO	RAM		<p><b>Outputs with Unreliable Values</b>  indicates whether the reading from the corresponding digital output is considered reliable.</p> <p>0=Reliable  1=Unreliable</p> <p>bit #0=BO1  bit #1=BO2  bit #2=BO3  bit #3=BO4  bit #4=BO5  bit #5=BO6  bit #6=BO7  bit #7=BO8  bit #8=BO9  bit #9=BO10  bit #10=BO11  bit #11=BO12</p>

## DIGITAL OUTPUTS 1-12, FB01-FB0C

Attribute	Data Type	Access	Storage	Default	Description
<b>AM</b>	FE	RW	NRAM	0	<b>Control Mode</b> specifies how the digital output is controlled.  0=Manual 1=Automatic
<b>AS</b>	FE	RO	RAM	0	<b>Alarm Status</b> indicates the alarm state for the input.  0=Normal 1=Contact 0>1 2=Contact 1>0 3=Change of state 8=Fire 10=Runtime 11=Supervisory
<b>CL</b>	E9	RW	NRAM		<b>Inrush Current Lockout Map</b> determines which digital outputs are considered in the same zone for the inrush current lockout
<b>CT</b>	FE	RW	NRAM	0	<b>Inrush Current Lockout Delay Timer</b> determines the amount of time to stage on digital outputs in the same inrush current lockout zone
<b>CV</b>	FE	RW	RAM	0	<b>Requested Output Value</b> specifies the current state of output. Because of various delays, this value may differ from <b>OU</b> .  0=Off 1=On
<b>FT</b>	FE	RW	NRAM		<b>Minimum Off Time</b> specifies the minimum time in second that must elapse after the output is turned off before it may be re-energized.
<b>GI</b>	FE	RW	NRAM	0	<b>GID of I/O Device</b> indicates the global identification number of the STATbus device associated with the digital output. Digital Outputs 1-6 will have a value of <b>GI</b> set equal to the serial number of the GPC. If Digital inputs 7-12 do not have a STATbus device mapped to them, <b>GI</b> for that input will be 0.
<b>IA</b>	DF	RW	NRAM		<b>[Additional Interlock] Input Attribute</b> specifies which input attribute (if any) will be used as an interlock for the associated output.
<b>IC</b>	DF	RW	NRAM		<b>[Additional Interlock] Input Channel</b> specifies which input channel (if any) will be used as an interlock for the associated output.

Attribute	Data Type	Access	Storage	Default	Description
IL	E9	RW	NRAM	0	<p><b>Inputs for Interlocking</b> specifies which inputs (if any) are used as interlocks for the associated output (a value of 1 indicates that that input is used for interlocking).</p> <p>bit #0=FE01 bit #1=FE02 bit #2=FE03 bit #3=FE04 bit #4=FE05 bit #5=FE06 bit #6=FE07 bit #7=FE08 bit #8=FE09 bit #9=FE0A bit #10=FE0B bit #11=FE0C bit #12=FE0D bit #13=FE0E bit #14=FE0F bit #15=FE10 bit #16=FE11 bit #17=FE12 bit #18=FE13 bit #19=FE14 bit #20=FE15 bit #21=FE16 bit #22=FE17 bit #23=FE18 bit #24=FE31 bit #25=FE32 bit #26=FE33 bit #27=FE34 bit #28=FE35 bit #29=FE36 bit #30=FE37 bit #31=FE38</p>
IP	FE	RW	EE	0	<p><b>Output Polarity.</b> allows the user to change the polarity of the input/output.</p> <p>0=Normal 1=Reverse</p>
IT	FE	RW	NRAM	0	<p><b>[Additional Interlock] Trigger</b> specifies the biasing of the trigger</p> <p>0=Trigger on zero 1=Trigger on non-zero</p>
NT	FE	RW	NRAM	0	<p><b>Minimum On Time</b> specifies the minimum time in seconds that must elapse after the output is energized before it may be turned off.</p>
ON	DF	RW	NRAM	"Digital Output n" where n=1 to 12	<p><b>Digital Output Name</b> specifies a user definable string that is used to help identify the channel or its function.</p>
OU	FE	RO	RAM	0	<p><b>Actual Output Value</b> specifies the actual state of the output.</p> <p>0=Off 1=On</p>
PW	FC	RW	NRAM	0	<p><b>Pulse Width when Output is On</b> specifies the "on" time (CV=1) in seconds (0.0 to 25.5) that the output should remain on after a transition from the off to on state.</p> <p>0=Disabled 0.1-25.5=pulse "on" duration in seconds</p>

Attribute	Data Type	Access	Storage	Default	Description
<b>RE</b>	FE	RO	RAM		<p><b>Data Reliability</b> indicates whether the reading from the corresponding digital output is considered reliable.</p> <p>0=Reliable 1=Unreliable</p>
<b>RH</b>	FC	RW	NRAM	0	<p><b>Run Hours</b> indicates the number of hours <b>CV</b>=1 for the input.</p>
<b>RL</b>	FC	RW	NRAM	0	<p><b>Run Hours Limit</b> specifies a number of hours that <b>CV</b> after which a run limit alarm is generated. Setting <b>RL</b>=0.0 disabled run limit alarms for the input.</p>
<b>SF</b>	FE	RO	RAM	0	<p><b>Schedule Feedback</b> indicates schedule feedback based on scheduling</p>
<b>SI</b>	FE	RW	NRAM	0	<p><b>Power-On Stagger Interval</b> specifies the time, in seconds, that must elapse after the controller is turned on or reset before the output can be turned on.</p>
<b>SM</b>	E9	RW	NRAM	0	<p><b>Schedules to Follow</b> enables scheduled alarm controlling for the associated digital output by selecting one or more of the available schedule control channels.</p> <p>0=Schedule disabled 1=Schedule enabled</p> <p><b>SM</b> is a bitmap with bit #0=F901 bit #1=F902 bit #2=F903 bit #3=F904 bit #4=F905 bit #5=F906 bit #6=F907 bit #7=F908 bit #8=SMARTStat1 bit #9=SMARTStat2 bit #10=SMARTStat3 bit #11=SMARTStat4 bit #12=SMARTStat5 bit #13=SMARTStat6 bit #14=SMARTStat7 bit #15=SMARTStat8 bit #16=SMARTStat9 bit #17=SMARTStat10 bit #18=SMARTStat11 bit #19=SMARTStat12 bit #20=Host Schedule bit #21=F900 bit #22=FC01</p>

## PID CONTROL 1-12, FA01-FA0C

Attribute	Data Type	Access	Storage	Default	Description
<b>AO</b>	FB	RO	RAM	0	<b>Analog Output Value</b> displays the output of the PID loop.
<b>CE</b>	FE	RW	NRAM	0	<b>Enable Control Loop?</b> enables/disables PID control.  0=No 1=Yes
<b>CS</b>	FB	RO	RAM	0	<b>Calculated Control Setpoint</b> specifies the effective (calculated) setpoint accounting for setup/ setback, manual setpoint adjustments, etc.
<b>DB</b>	FB	RW	NRAM	0	<b>Desired Control Deadband</b> specifies the deadband that is used to control cycling around the setpoint. If the current value of the input channel is between <b>SP-(DB/2)</b> and <b>SP+(DB/2)</b> , the measured variable is considered to be at its setpoint.
<b>DL</b>	FB	RO	NRAM		<b>Demand Load</b> indicates the demand load for the control loop. This is the amount by which <b>CS</b> differs from the loop measured variable specified in <b>IC</b> and <b>IA</b> .
<b>FZ</b>	FE	RW	NRAM	0	<b>Force Loop to 0% when Disabled</b> specifies if the loop should be forced to zero or stay at <b>PO</b> when the loop is disabled
<b>IA</b>	DF	RW	NRAM		<b>Loop Measured Attribute</b> specifies the attribute associated with the channel specified in <b>IC</b> to be used as the measured variable for PID control loop.
<b>IC</b>	DF	RW	NRAM		<b>Loop Measured Channel</b> specifies the input channel to be used as the measured variable for the PID control loop.
<b>MR</b>	FD	RW	NRAM	0	<b>Maximum Amount to Reset Setpoint</b> the maximum amount to reset the control loop setpoint.
<b>OH</b>	FB	RW	NRAM	100	<b>Output High Limit</b> defines the maximum output for the PID control loop.
<b>OL</b>	FB	RW	NRAM	0	<b>Output Low Limit</b> defines the minimum output for the PID control loop.
<b>ON</b>	DF	RW	NRAM	"PID Control n" where n=1 to 12	<b>Control Loop Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>PB</b>	FD	RW	NRAM	0	<b>Proportional Control Band</b> specifies a range, centered around the loop setpoint <b>SP</b> , where the output signal is proportional. If the value of the selected input channel is outside the proportional band, the proportional component of the PID calculation is clamped at <b>OL</b> or <b>OH</b> as appropriate.
<b>PO</b>	FB	RW	NRAM	0	<b>Percent Output Value</b> displays the calculated output of the PID control loop. <b>PO</b> ranges from <b>OL</b> to <b>OH</b> .
<b>P0</b>	FE	RW	NRAM	0	<b>Permanent Stat Override Enabled</b> allows Stat overrides to become permanent.

Attribute	Data Type	Access	Storage	Default	Description
<b>RA</b>	DF	RW	NRAM	00	<b>Reset Attribute</b> specifies the attribute associated with the channel specified in <b>RC</b> to be used as the reset variable for the PID control loop.
<b>RC</b>	DF	RW	NRAM	0	<b>Reset Channel</b> specifies the input channel to be used as the reset variable for the PID control loop.
<b>RL</b>	FD	RW	NRAM	0	<b>Limit for Maximum Reset</b> specifies the reset limit of the control loop. When <b>RV</b> reaches a value of <b>RL</b> , the control loop setpoint will be reset by the maximum amount <b>RV</b> .
<b>RP</b>	FE	RW	NRAM	0	<b>Reset Period</b> specifies a time, in seconds (0 to 65,535) over which the output of the control loop should be adjusted (reset) using integral action.  0=Disabled 1 to 65,535=Integral reset period, in seconds
<b>RS</b>	FD	RW	NRAM	0	<b>Setpoint at which Reset Action Begins</b> specifies the setpoint of the control loop at which reset action begins.
<b>RT</b>	FD	RW	NRAM	0	<b>Derivative Rate</b> specifies a a percentage of the amount of derivative error that is contributed each second to the PID output of the control loop (0.0 to 25.5%).  0.0=Disable 0.1 to 25.5=Derivative rate in %/second
<b>SG</b>	FE	RW	NRAM	0	<b>Control Action</b> specifies whether the controller's output should be increased or decreased when error is positive.  0=Normal (increase for positive error) 1=Reverse (decrease for positive error)
<b>SM</b>	E9	RW	NRAM	0	<b>Schedules to Follow</b> enables scheduled alarm controlling for the associated PID control loop by selecting one or more of the available schedule control channels.  0=Schedule disabled 1=Schedule enabled  <b>SM</b> is a bitmap with bit #0=F901 bit #1=F902 bit #2=F903 bit #3=F904 bit #4=F905 bit #5=F906 bit #6=F907 bit #7=F908 bit #8=SMARTStat1 bit #9=SMARTStat2 bit #10=SMARTStat3 bit #11=SMARTStat4 bit #12=SMARTStat5 bit #13=SMARTStat6 bit #14=SMARTStat7 bit #15=SMARTStat8 bit #16=SMARTStat9 bit #17=SMARTStat10 bit #18=SMARTStat11 bit #19=SMARTStat12 bit #20=Host Schedule bit #21=F900 bit #22=FC01

Attribute	Data Type	Access	Storage	Default	Description
<b>SP</b>	FB	RW	NRAM	0	<b>Loop Setpoint</b> specifies the desired value of the variable selected in <b>IC</b> and <b>IA</b> .
<b>SR</b>	FE	RW	NRAM	100	<b>Soft Start Ramp</b> specifies the maximum percentage change per minute for the associated output under the following conditions: when the controller is initially powered up or reset; upon transitions from unoccupied to occupied mode and upon cancellation of an interlock failure or fire condition.
<b>SU</b>	FD	RW	NRAM	0	<b>Unoccupied Setup/Setback</b> specifies a value (0.0 to 25.5) which is added to (if <b>SG=1</b> ) or subtracted from (if <b>SG=0</b> ) the control loop setpoint during scheduled unoccupied periods.
<b>SV</b>	FE	RO	RAM	4	<b>Current Schedule Value</b> indicates the current value of the controlling schedule  0=Unoccupied 1=Warmup 2=Occupied 3=Night setback 4=Not Set/Disabled
<b>TS</b>	FD	RO	NRAM		<b>Thermostat SP Adjustment</b> indicates any setpoint adjustments read in from a SmartSTAT.

THERMOSTATIC CONTROL 1-12, FA11-FA1C

Attribute	Data Type	Access	Storage	Default	Description
<b>CE</b>	FE	RW	NRAM	0	<b>Enable Control Loop?</b> enables/disables thermostatic control for the associated control loop.  0=Disabled 1=Enabled
<b>CS</b>	FB	RO	RAM	0	<b>Calculated Control Setpoint</b> specifies the calculated (actual) control setpoint that is used by the thermostatic control loop. <b>CS</b> accounts for the effects of setup/setback ( <b>SU</b> ) during scheduled unoccupied periods.
<b>CV</b>	FE	RW	RAM	0	<b>Current Value</b> indicates the current desired state of the output(s) selected in <b>OB</b> .
<b>DB</b>	FB	RW	NRAM	0	<b>Desired Control DeadBand</b> specifies a control hysteresis that is used to keep <b>CV</b> from toggling when the value is on the border between two states.
<b>DL</b>	FB	RO	NRAM		<b>Demand Load</b> indicates the demand load for the control loop. This is the amount by which <b>CS</b> differs from the loop measured variable specified in <b>IC</b> and <b>IA</b> .
<b>IA</b>	DF	RW	NRAM		<b>Loop Measured Attribute</b> specifies the attribute associated with the channel specified in <b>IC</b> to be used as the measured variable for Thermostatic control loop.
<b>IC</b>	DF	RW	NRAM		<b>Loop Measured Channel</b> specifies the input channel to be used as the measured variable for the Thermostatic control loop.
<b>OB</b>	E9	RW	NRAM	0	<b>Output Bitmap</b> specify which digital outputs will be controlled by the thermostatic control loop.  bit #0=BO1 bit #1=BO2 bit #2=BO3 bit #3=BO4 bit #4=BO5 bit #5=BO6 bit #6=BO7 bit #7=BO8 bit #8=BO9 bit #9=BO10 bit #10=BO11 bit #11=BO12
<b>ON</b>	DF	RW	NRAM	"Thermostatic Control n" where n=1 to 4	<b>Thermostatic Control Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>P0</b>	FE	RW	NRAM	0	<b>Permanent Stat Override Enabled</b> allows Stat overrides to become permanent.
<b>RA</b>	DF	RW	NRAM		<b>Reset Attribute</b> specifies the attribute associated with the channel specified in <b>IC</b> to be used for thermostatic control.
<b>RC</b>	DF	RW	NRAM		<b>Reset Channel</b> specifies the channel to be used for thermostatic control.

Attribute	Data Type	Access	Storage	Default	Description
<b>SG</b>	FE	RW	NRAM	0	<p><b>Control Action</b> specifies the control sign for the thermostatic control application.</p> <p>0=Cooling 1=Heating</p>
<b>SM</b>	E9	RW	NRAM	0	<p><b>Schedules to Follow</b> enables scheduled alarm controlling for the thermostatic control loop by selecting one or more of the available schedule control channels.</p> <p>0=schedule disabled 1=schedule enabled</p> <p><b>SM</b> is a bitmap with bit #0=F901 bit #1=F902 bit #2=F903 bit #3=F904 bit #4=F905 bit #5=F906 bit #6=F907 bit #7=F908 bit #8=SMARTStat1 bit #9=SMARTStat2 bit #10=SMARTStat3 bit #11=SMARTStat4 bit #12=SMARTStat5 bit #13=SMARTStat6 bit #14=SMARTStat7 bit #15=SMARTStat8 bit #16=SMARTStat9 bit #17=SMARTStat10 bit #18=SMARTStat11 bit #19=SMARTStat12 bit #20=Host Schedule bit #21=F900 bit #22=FC01</p>
<b>SP</b>	FB	RW	NRAM	0	<p><b>Loop Setpoint</b> specifies the desired setpoint for the thermostatic control loop.</p>
<b>SU</b>	FB	RW	NRAM	0	<p><b>Unoccupied Setup/Setback</b> specifies a value (0.0 to 25.5) which is added to the control setpoint during scheduled unoccupied periods.</p>
<b>SV</b>	FE	RO	RAM	4	<p><b>Current Schedule Value</b> indicates the current value of the controlling schedule</p> <p>0=Unoccupied 1=Warmup 2=Occupied 3=Night setback 4=Not Set/Disabled</p>
<b>TS</b>	FD	RO	NRAM		<p><b>Thermostat SP Adjustment</b> indicates any setpoint adjustments read in from a SmartSTAT.</p>

FLOATING POINT CONTROL 1-2, FA21-FA22

Attribute	Data Type	Access	Storage	Default	Description
<b>CE</b>	FE	RW	NRAM	0	<b>Enable Control Loop?</b> enables/disables floating point control for the associated control loop.  0=No 1=Yes
<b>CF</b>	FE	RW	NRAM	0	<b>Communication Failure Enable?</b> specifies what action to take in the event that a communication failure is detected.  0=No 1=Yes
<b>CP</b>	FB	RO	RAM	0	<b>Current Position</b> indicates the current position of the motor.
<b>CR</b>	FE	RW	NRAM	0	<b>Motor Creep Function</b> specifies how the controller handles automatic calibrations at minimum and maximum positions.  0=Drive motor constantly if <b>DP</b> =0% or <b>DP</b> =100% 1=Creep motor output by 1% per minute if <b>DP</b> =0% or <b>DP</b> =100%
<b>CS</b>	FB	RO	RAM	0	<b>Calculated Control Setpoint</b> indicates the calculated control setpoint. This value accounts for any reset or setup/setback action on the loop setpoint.
<b>DB</b>	FB	RW	NRAM	0	<b>Desired Control Deadband</b> specifies the deadband that is used to control cycling around the setpoint. If the current value of the input channel is between <b>SP</b> -( <b>DB</b> /2) and <b>SP</b> +( <b>DB</b> /2), the measured variable is considered to be at its setpoint
<b>DL</b>	FB	RO	NRAM		<b>Demand Load</b> indicates the demand load for the control loop. This is the amount by which <b>CS</b> differs from the loop measured variable specified in <b>IC</b> and <b>IA</b> .
<b>DO</b>	FE	RW	NRAM	0	<b>Digital Output Pair Selection</b> specifies which Digital Output pair is to be used for floating point control.  0=none 1=D01+D02 2=D03+4 3=D05+6 4=D07+8 5=D09+10 6=D011+12
<b>DP</b>	FB	RW	NRAM	0	<b>Desired Position</b> specifies the desired output position (0-100%) of the associated motor.
<b>FI</b>	FB	RW	NRAM	0	<b>Fire Position</b> specifies the failure position (0-100%) to use when a fire event is detected.
<b>FP</b>	FB	RW	NRAM	0	<b>Interlock/Comm Failure Position</b> specifies the failure position (0-100%) to use when an input interlock failure occurs. <b>FP</b> is used when the current value of any of the inputs specified by <b>IL</b> has a value of 1.
<b>IA</b>	DF	RW	NRAM		<b>Input Attribute</b> specifies the channel to be used for floating point control.

Attribute	Data Type	Access	Storage	Default	Description
<b>IC</b>	DF	RW	NRAM		<b>Input Channel</b> specifies the attribute associated with the channel specified in <b>IC</b> to be used for floating point control.
<b>IL</b>	E9	RW	NRAM	0	<b>Input for Interlock</b> specifies which inputs are to be used for interlocking of the associated floating point control loop.  <b>IL</b> is a bitmap with bit 1 corresponding to IUI1, bit 2=UI2, etc. up to bit 12=UIO.
<b>MR</b>	FB	RW	NRAM	0	<b>Maximum Amount to Reset Setpoint</b> specifies the maximum amount o reset <b>SP</b> .
<b>ON</b>	DF	RW	NRAM	"Floating Point Control n" where n=1 to 2	<b>Floating Point Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>PB</b>	FB	RW	NRAM	0	<b>Proportional Control Band</b> specifies a range, centered around the loop setpoint <b>SP</b> , where the output signal is proportional.
<b>PE</b>	FE	RW	NRAM	0	<b>Floating Point Controlled Pair Enable</b> specifies whether the motor is to be controlled using pulsed pairs.  0=No 1=Yes
<b>P0</b>	FE	RW	NRAM	0	<b>Permanent Stat Override Enabled</b> allows Stat overrides to become permanent.
<b>RA</b>	DF	RW	NRAM		<b>Reset Attribute</b> specifies the attribute associated with the channel specified in <b>RC</b> to determine reset.
<b>RC</b>	DF	RW	NRAM		<b>Reset Channel</b> specifies the channel to be used to determine reset.
<b>RI</b>	FE	RW	NRAM	0	<b>Motor Recalibrate Interval</b> specifies a time interval in hours (0-255) the defines how often the associated floating point control loop is recalibrated.  0=Calibration disabled <b>RI</b> > 0 =Recalibrate every <b>RI</b> hours
<b>RL</b>	FB	RW	NRAM	0	<b>Limit for Maximum Reset</b> specifies the reset limit of the control loop. When the reset variable specified in <b>RV</b> and <b>RA</b> reaches a value of <b>RL</b> , the control loop setpoint will be reset by the maximum amount <b>MR</b> .
<b>RP</b>	FE	RW	NRAM	0	<b>Reset Period</b> specifies a time, in seconds (0 to 65,535) over which the output of the control loop should be adjusted (reset).  0=Disabled 1 to 65,535=Reset period, in seconds
<b>RS</b>	FB	RW	NRAM	0	<b>Setpoint at which Reset Action Begins</b> specifies the setpoint of the control loop at which reset action begins.
<b>SG</b>	FE	RW	NRAM	0	<b>Control Action</b> specifies whether the controller's output should be increased or decreased when the control signal is positive.  0=Normal (increase for positive error) 1=Reverse (decrease for positive error)

Attribute	Data Type	Access	Storage	Default	Description
SM	E9	RW	NRAM	0	<p><b>Schedules to Follow</b> enables scheduled alarm controlling by selecting one or more of the available schedule control channels.</p> <p>0=Schedule disabled 1=Schedule enabled</p> <p><b>SM</b> is a bitmap with bit #0=F901 bit #1=F902 bit #2=F903 bit #3=F904 bit #4=F905 bit #5=F906 bit #6=F907 bit #7=F908 bit #8=SMARTStat1 bit #9=SMARTStat2 bit #10=SMARTStat3 bit #11=SMARTStat4 bit #12=SMARTStat5 bit #13=SMARTStat6 bit #14=SMARTStat7 bit #15=SMARTStat8 bit #16=SMARTStat9 bit #17=SMARTStat10 bit #18=SMARTStat11 bit #19=SMARTStat12 bit #20=Host Schedule bit #21=F900 bit #22=FC01</p>
SP	FB	RW	NRAM	0	<p><b>Loop Setpoint</b> specifies the desired setpoint for the floating point control loop.</p>
SU	FB	RW	NRAM	0	<p><b>Unoccupied Setup/Setback</b> specifies a value (0.0 to 25.5) which is added to (if <b>SG</b>=1) or subtracted from (if <b>SG</b>=0) the control loop setpoint during scheduled unoccupied periods</p>
SV	FE	RO	RAM	4	<p><b>Current Schedule Value</b> indicates the current value of the controlling schedule</p> <p>0=Unoccupied 1=Warmup 2=Occupied 3=Night setback 4=Not Set/Disabled</p>
TS	FD	RO	NRAM		<p><b>Thermostat SP Adjustment</b> indicates any setpoint adjustments read in from a SmartSTAT.</p>
TT	FE	RW	NRAM	0	<p><b>Motor Travel Time</b> specifies the time, in seconds (0-3000), that it takes the motor to move from its fully closed to its fully open positions.</p>

## SCHEDULE SUMMARY, F900

Attribute	Data Type	Access	Storage	Default	Description
<b>AS</b>	E9	RO	RAM		<p><b>Active Schedules</b> lists the schedules which are currently active.</p> <p>bit #0=F901 bit #1=F902 bit #2=F903 bit #3=F904 bit #4=F905 bit #5=F906 bit #6=F907 bit #7=F908</p>
<b>C1</b>	FE	RO	RAM		<p><b>Schedule 1</b> indicates the current state of Schedule 1.</p> <p>0=Unoccupied 1=Warm-up 2=Occupied 3=Night setback</p>
<b>C2</b>	FE	RO	RAM		<p><b>Schedule 2</b> indicates the current state of Schedule 2</p> <p>0=Unoccupied 1=Warm-up 2=Occupied 3=Night setback</p>
<b>C3</b>	FE	RO	RAM		<p><b>Schedule 3</b> indicates the current state of Schedule 3.</p> <p>0=Unoccupied 1=Warm-up 2=Occupied 3=Night setback</p>
<b>C4</b>	FE	RO	RAM		<p><b>Schedule 4</b> indicates the current state of Schedule 4.</p> <p>0=Unoccupied 1=Warm-up 2=Occupied 3=Night setback</p>
<b>C5</b>	FE	RO	RAM		<p><b>Schedule 5</b> indicates the current state of Schedule 5.</p> <p>0=Unoccupied 1=Warm-up 2=Occupied 3=Night setback</p>
<b>C6</b>	FE	RO	RAM		<p><b>Schedule 6</b> indicates the current state of Schedule 6.</p> <p>0=Unoccupied 1=Warm-up 2=Occupied 3=Night setback</p>
<b>C7</b>	FE	RO	RAM		<p><b>Schedule 7</b> indicates the current state of Schedule 7.</p> <p>0=Unoccupied 1=Warm-up 2=Occupied 3=Night setback</p>

Attribute	Data Type	Access	Storage	Default	Description
<b>C8</b>	FE	RO	RAM		<p><b>Schedule 8</b> indicates the current state of Schedule 8.</p> <p>0=Unoccupied 1=Warm-up 2=Occupied 3=Night setback</p>
<b>CV</b>	FE	RO	RAM		<p><b>Current Schedule Value</b> indicates the current state of the controller.</p> <p>0=Unoccupied 1=Warm-up 2=Occupied 3=Night setback</p>
<b>DH</b>	FE	RW	NRAM	0	<p><b>Holiday?</b> specifies whether today is one of the holidays specified in <b>H0-H9</b>.</p>
<b>FB</b>	FE	RO	NRAM		<p><b>Main Schedule Set By</b> indicates what is controlling the main schedule</p> <p>0=Extended Occupancy 1=Host Override 2=Broadcast 3=Local Schedule 4=Inactive State</p>
<b>H0</b>	E4	RW	NRAM	0	<b>Programmed Holiday</b>
<b>H1</b>	E4	RW	NRAM	0	<b>Programmed Holiday</b>
<b>H2</b>	E4	RW	NRAM	0	<b>Programmed Holiday</b>
<b>H3</b>	E4	RW	NRAM	0	<b>Programmed Holiday</b>
<b>H4</b>	E4	RW	NRAM	0	<b>Programmed Holiday</b>
<b>H5</b>	E4	RW	NRAM	0	<b>Programmed Holiday</b>
<b>H6</b>	E4	RW	NRAM	0	<b>Programmed Holiday</b>
<b>H7</b>	E4	RW	NRAM	0	<b>Programmed Holiday</b>
<b>H8</b>	E4	RW	NRAM	0	<b>Programmed Holiday</b>
<b>H9</b>	E4	RW	NRAM	0	<b>Programmed Holiday</b>
<b>HE</b>	FE	RW	NRAM	0	<p><b>Enable Host Schedule</b> specifies whether to use the schedule broadcast by the host.</p> <p>0=No 1=Yes</p>
<b>HO</b>	FE	RW	NRAM	0	<p><b>Host Schedule Value</b> specifies the desired host schedule override state.</p> <p>0=Unoccupied 1=Warmup 2=Occupied 3=Night setback</p>
<b>IS</b>	FE	RW	NRAM	0	<p><b>Inactive State</b> the schedule mode that the controller will default to if there are no schedules which are currently active.</p>

Attribute	Data Type	Access	Storage	Default	Description
<b>ON</b>	DF	RW	NRAM	"Schedule Summary"	<b>Schedule Summary Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>SO</b>	E9	RO	NRAM	0	<b>Digital Stat Override Status</b> displays the state of each STAT's local override flag.  bit #0=SMARTStat 1 bit #1=SMARTStat 2 bit #2=SMARTStat 3 bit #3=SMARTStat 4 bit #4=SMARTStat 5 bit #5=SMARTStat 6 bit #6=SMARTStat 7 bit #7=SMARTStat 8 bit #8=SMARTStat 9 bit #9=SMARTStat 10 bit #10=SMARTStat 11 bit #11=SMARTStat 12
<b>TH</b>	FE	RO	NRAM	0	<b>Tomorrow's Holiday Status</b> specifies whether tomorrow is one of the holidays specified in H0-H9.

SCHEDULES 1-8, F901-F908

Attribute	Data Type	Access	Storage	Default	Description
AD	E9	RW	NRAM	0	<p><b>Active Days</b>                      bitmap that specifies which days the schedule is active. Setting the appropriate bit to 1 indicates the schedule will be active on that day.</p> <p>bit #0=Monday                      bit #1=Tuesday                      bit #2=Wednesday                      bit #3=Thursday                      bit #4=Friday                      bit #5=Saturday                      bit #6=Sunday                      bit #7=Holiday</p>
AN	FE	RO	NRAM		<p><b>Schedule is Active Now</b>                      indicates if today is an actively scheduled day</p> <p>0=No                      1=Yes</p>
AT	FE	RO	NRAM		<p><b>Schedule Will Be Active Tomorrow</b>                      indicates if tomorrow is an actively scheduled day</p> <p>0=No                      1=Yes</p>
AO	FE	RW	NRAM	0	<p><b>All Day Override</b>                      set the schedule state for the entire day.</p> <p>0=None                      1=Unoccupied                      2=Warmup                      3=Occupied                      4=Night Setback</p>
AY	FE	RO	NRAM		<p><b>Schedule Was Active Yesterday</b>                      indicates if yesterday was an actively scheduled day</p> <p>0=No                      1=Yes</p>
CO	FE	RO	NRAM		<p><b>Currently Occupied Flag</b>                      indicates whether the current schedule value is currently occupied</p> <p>0=No                      1=Yes</p>
CS	FE	RO	NRAM		<p><b>Currently in Setback Flag</b>                      indicates whether the current schedule value is Night Setback</p> <p>0=No                      1=Yes</p>
CU	FE	RO	NRAM		<p><b>Currently Unoccupied Flag</b>                      indicates whether the current schedule value is Unoccupied</p> <p>0=No                      1=Yes</p>
CV	FE	RO	RAM	0	<p><b>Current Schedule Value</b>                      indicates the current state of the schedule.</p> <p>0=Unoccupied                      1=Warm-up                      2=Occupied                      3=Night Setback</p>

Attribute	Data Type	Access	Storage	Default	Description
<b>CW</b>	FE	RO	NRAM		<b>Currently In Warmup Flag</b> indicates whether the current schedule value is in Warmup  0=No 1=Yes
<b>DA</b>	FE	RO	NRAM		<b>Current Day of Week</b> shows the current day of the week  0=Mon 1=Tues 2=Wed 3=Thurs 4=Fri 5=Sat 6=Sun
<b>FB</b>	FE	RO	NRAM		<b>Schedule Feedback</b> provides feedback text on the current schedule status  1=Inactive Today 2=Inactive Holiday Today 3=All Day Override 4=Value at Midnight 5=Tracking Time Entry
<b>NS</b>	E6	RW	NRAM	19:00	<b>Time to go Night Setback</b> specifies the time when night setback mode should begin.
<b>OC</b>	E6	RW	NRAM	08:00	<b>Time to go Occupied</b> specifies the time when occupied mode should begin.
<b>ON</b>	DF	RW	NRAM	"Schedule n" where n=1 to 8	<b>Schedule Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>SD</b>	FE	RW	NRAM	0	<b>Schedule Value at Midnight(Unitl First Entry)</b> On active days the schedule's Current Value is set to the Schedule Value at midnight  0=Unoccupied 1=Warmup 2=Occupied 3=Night Setback
<b>SF</b>	FE	RO	NRAM		<b>Currently Off Flag</b> indicates whether the current schedule value is in Night Setback or Unoccupied  0=No 1=Yes
<b>SO</b>	FE	RO	NRAM		<b>Currently On Flag</b> indicates whether the current schedule value is in Warmup or Occupied  0=No 1=Yes
<b>TM</b>	E6	RO	NRAM		<b>Current Controller Time</b> indicates the current time in HH:MM format
<b>UN</b>	E6	RW	NRAM	17:00	<b>Time to go Unoccupied</b> specifies the time when unoccupied mode should begin.
<b>WO</b>	E6	RW	NRAM	07:30	<b>Time to go Warm-Up</b> specifies the time when warm-up mode should begin.

TIMERS 1, F800

Attribute	Data Type	Access	Storage	Default	Description
<b>C0 - C9</b>	FE	RO	RAM	0	<b>Timer 0-9 Counter (Seconds)</b> reflects the current state of the counter.
<b>M0-M9</b>	FE	RW	NRAM	0 = Disabled	<b>Timer 0-9 Mode</b> Defines the configured mode for the corresponding timer.  0=Disabled 1=Simple Downcounter (No clearing -w- 0) 2=Simple Downcounter (Clearing with 0 allowed) 3=Triggered Downcounter (Any positive # restarts counter) 4=Triggered Downcounter (Any positive # restarts & 0 clears) 5=Simple Upcounter 6=Simple Clearable Upcounter (accepts 0 only)
<b>ON</b>	DF	RW	NRAM	Timers Channel	<b>Channel Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>V0-V9</b>	FE	RW	NRAM	0	<b>Timer 0-9 Reset Value (in seconds)</b> defines the reset value for the corresponding timer.

## SCALES 1-12, F711-F71C

Attribute	Data Type	Access	Storage	Default	Description
<b>CV</b>	FC	RO	RAM		<b>Current Value</b> specifies the calculated scaled value.
<b>DT</b>	FE	RW	NRAM	253	<b>PUP Datatype for Values</b> specifies the PUP datatype for the scale.
<b>IA</b>	DF	RW	NRAM	0	<b>Input Attribute</b> specifies the attribute associated with the channel specified in <b>IC</b> to be scaled.
<b>IC</b>	DF	RW	NRAM		<b>Input Channel</b> specifies the channel to be scaled.
<b>IV</b>	FD	RO	NRAM	0	<b>Input's Current Value</b> indicates the current value of the associated input
<b>ON</b>	DF	RW	NRAM	"Scale n" where n=1 to 12	<b>Scale Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>X1</b>	FE	RW	NRAM	0	<b>Input range X1 value</b> specifies the minimum value of the input.
<b>X2</b>	FE	RW	NRAM	0	<b>Input range X2 value</b> specifies the maximum value of the input.
<b>Y1</b>	FE	RW	NRAM	0	<b>Output range Y1 value</b> specifies the minimum value of the scaled output.
<b>Y2</b>	FE	RW	NRAM	0	<b>Output range Y2 value</b> specifies the maximum value of the scaled output.

PIECEWISE CURVES 1-8, F701-F708

Attribute	Data Type	Access	Storage	Default	Description
<b>DT</b>	FE	RW	NRAM	253	<b>PUP Datatype</b> specifies the PUP datatype for the piecewise curve.
<b>ON</b>	DF	RW	NRAM	"Piecewise Curve n" where n=1 to 8	<b>Piecewise Curves Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>X1</b>	FE	RW	NRAM	0	<b>Point 1's value in raw counts</b> specifies the x coordinate of point 1.
<b>X2</b>	FE	RW	NRAM	0	<b>Point 2's value in raw counts</b> specifies the x coordinate of point 2.
<b>X3</b>	FE	RW	NRAM	0	<b>Point 3's value in raw counts</b> specifies the x coordinate of point 3.
<b>X4</b>	FE	RW	NRAM	0	<b>Point 4's value in raw counts</b> specifies the x coordinate of point 4.
<b>X5</b>	FE	RW	NRAM	0	<b>Point 5's value in raw counts</b> specifies the x coordinate of point 5.
<b>X6</b>	FE	RW	NRAM	0	<b>Point 6's value in raw counts</b> specifies the x coordinate of point 6.
<b>X7</b>	FE	RW	NRAM	0	<b>Point 7's value in raw counts</b> specifies the x coordinate of point 7.
<b>X8</b>	FE	RW	NRAM	0	<b>Point 8's value in raw counts</b> specifies the x coordinate of point 8.
<b>X9</b>	FE	RW	NRAM	0	<b>Point 9's value in raw counts</b> specifies the x coordinate of point 9.
<b>XA</b>	FE	RW	NRAM	0	<b>Point 10's value in raw counts</b> specifies the x coordinate of point 10.
<b>XB</b>	FE	RW	NRAM	0	<b>Point 11's value in raw counts</b> specifies the x coordinate of point 11.
<b>Y1</b>	FE	RW	NRAM	0	<b>Point 1's value in engineering units</b> specifies the y coordinate of point 1.
<b>Y2</b>	FE	RW	NRAM	0	<b>Point 2's value in engineering units</b> specifies the y coordinate of point 2.
<b>Y3</b>	FE	RW	NRAM	0	<b>Point 3's value in engineering units</b> specifies the y coordinate of point 3.
<b>Y4</b>	FE	RW	NRAM	0	<b>Point 4's value in engineering units</b> specifies the y coordinate of point 4.
<b>Y5</b>	FE	RW	NRAM	0	<b>Point 5's value in engineering units</b> specifies the y coordinate of point 5.
<b>Y6</b>	FE	RW	NRAM	0	<b>Point 6's value in engineering units</b> specifies the y coordinate of point 6.
<b>Y7</b>	FE	RW	NRAM	0	<b>Point 7's value in engineering units</b> specifies the y coordinate of point 7.

---

Attribute	Data Type	Access	Storage	Default	Description
<b>Y8</b>	FE	RW	NRAM	0	<b>Point 8's value in engineering units</b> specifies the y coordinate of point 8.
<b>Y9</b>	FE	RW	NRAM	0	<b>Point 9's value in engineering units</b> specifies the y coordinate of point 9.
<b>YA</b>	FE	RW	NRAM	0	<b>Point 10's value in engineering units</b> specifies the y coordinate of point 10.
<b>YB</b>	FE	RW	NRAM	0	<b>Point 11's value in engineering units</b> specifies the y coordinate of point 11.

PERSISTED/REMAP VALUES 1-64, F601-F640

<b>A1</b>	DF	RW	NRAM		<b>Input Attribute 1</b> specifies the attribute of the input to be used as input 1
<b>A2</b>	DF	RW	NRAM		<b>Input Attribute 2</b> specifies the attribute of the input to be used as input 2
<b>BC</b>	FE	RW	NRAM	0	<b>Clear Data At Boot</b> specifies whether data is cleared at boot  0=No(Retain the Data) 1=Yes(Clear Upon Boot)
<b>C1</b>	DF	RW	NRAM		<b>Input Channel 1</b> specifies the channel of the input to be used as input 1.
<b>C2</b>	DF	RW	NRAM		<b>Input Channel 2</b> specifies the channel of the input to be used as input 2
<b>CM</b>	FE	RW	NRAM	0	<b>Datatype Coercion Mode</b> determines where the datatype is set  0=As specified by Input 1=As specified here
<b>CV</b>	FE	RW	RAM	0	<b>Current Value</b> <b>specifies the current value of the associated persisted value</b>
<b>DT</b>	FE	RW	NRAM	254	<b>PUP Datatype for CV</b> indicates the datatype of the current value of the associated persisted value
<b>FB</b>	FE	RO	NRAM	0	<b>Feedback Status</b> provides feedback status for the channel  0=Everything is working fine 1=Can't read the input 2=Can't coerce the input data 3=Can't write to the output 4=Can't coerce the output data
<b>OA</b>	DF	RW	NRAM		<b>Output Attribute</b> specifies the attribute for the output
<b>OC</b>	DF	RW	NRAM		<b>Output Channel</b> specifies the channel of the output
<b>OM</b>	FE	RW	NRAM		<b>Operating Mode</b> specifies the operating mode for the persisted value  0=Static Value 1=Pull Data In and Push Data Out 2=Pull Data In 3=Push Data Out
<b>ON</b>	DF	RW	NRAM	"Value n" where n=1 to 64	<b>Channel Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>TA</b>	DF	RW	NRAM		<b>Trigger Attribute</b> specifies the attribute of the input to be used as a trigger
<b>TC</b>	DF	RW	NRAM		<b>Trigger Channel</b> specifies the channel of the input to be used as a trigger

<b>TM</b>	FE	RW	NRAM	0	<b>Trigger Mode</b> specifies the mode to be used when triggered  0=No Trigger 1=Read/Write Data only when triggered 2= R/W I2 when triggered (else I1) 3=R/W I1 when triggered (else I2)
-----------	----	----	------	---	---

ENTHALPY 1-4, F501- F504

Attribute	Data Type	Access	Storage	Default	Description
CV	F8	RO	RAM		<b>Current Value</b> indicates the result of the calculation  English = Btu's per lb of dry air Metric = Kilojoules per kg of dry air
DT	FE	RW	NRAM	248	<b>PUP Datatype</b> indicates the datatype of the current value of the associated enthalpy channel
HA	DF	RW	NRAM		<b>Humidity Input Attribute</b> specifies the attribute of the input to be used as the humidity input
HC	DF	RW	NRAM		<b>Humidity Input Channel;</b> specifies the channel of the input to be used as the humidity input
HV	F8	RO	NRAM	0	<b>%Relative Humidity Value</b> specifies the current value of the humidity input
ON	DF	RW	NRAM	Enthalpy n" where n=1 to 4	<b>Enthalpy Name</b> specifies a user definable string that is used to help identify the channel or its function
TA	DF	RW	NRAM		<b>Temperature Input Attribute</b> specifies the attribute of the input to be used as the temperature input
TC	DF	RW	NRAM		<b>Temperature Input Channel</b> specifies the channel of the input to be used as the temperature input
TV	F8	RO	NRAM	0	<b>Dry Bulb Temperature Value</b> specifies the current value of the temperature input

## STAGING 1-6 F401-F406

Attribute	Data Type	Access	Storage	Default	Description
<b>AI</b>	DF	RW	NRAM		<b>Interlock Attribute</b> specifies the attribute of the input to be used for interlocking
<b>DT</b>	FE	RW	NRAM	249	<b>PUP Datatype</b> indicates the datatype of the current value of the associated enthalpy channel
<b>ON</b>	DF	RW	NRAM	"Staging n" where n= 1-6	<b>Staging Name</b> specifies a user definable string that is used to help identify the channel or its function
<b>SM</b>	FE	RW	NRAM	0	<b>Staging Mode</b> specifies which staging mode should be used  0=Channel Disabled 1=Delay On/Delay Off 2=Threshold Based Staging
<b>LM</b>	FE	RW	NRAM	0	<b>Lead/Lag/Leveling Mode</b> specifies the lead/lag/leveling mode  0=Normal (First on Last Off 1= Automatic Wear Leveling
<b>NS</b>	FE	RW	NRAM	2	<b>Number of Stages &lt;Max Loading&gt;</b> determines the number of stages associated with this channel
<b>IC</b>	DF	RW	NRAM		<b>Input Channel</b> specifies the channel of the input to be used for this channel
<b>IA</b>	DF	RW	NRAM		<b>Input Attribute</b> specifies the attribute of associated with the channel specified in IC
<b>IV</b>	F9	RO	NRAM	0.000	<b>Input's Current Value</b> the current value of the input associated with this channel
<b>II</b>	FE	RW	NRAM	0	<b>Invert the Input</b> specifies whether the input should be inverted
<b>IS</b>	FE	RW	NRAM	0	<b>Invert the Setpoints? &lt;Higher Stages = Lower Numbers&gt;</b> reverses the action for the channel
<b>P1</b>	F9	RW	NRAM	0.000	<b>Setpoint 1 (Unload point in Delay On/Delay Off Mode)</b> specifies the value for setpoint 8
<b>P2</b>	F9	RW	NRAM	0.000	<b>Setpoint 2 (Unload point in Delay On/Delay Off Mode)</b> specifies the value for setpoint 8
<b>P3</b>	F9	RW	NRAM	0.000	<b>Setpoint 3</b> specifies the value for setpoint 8
<b>P4</b>	F9	RW	NRAM	0.000	<b>Setpoint 4</b> specifies the value for setpoint 8
<b>P5</b>	F9	RW	NRAM	0.000	<b>Setpoint 5</b> specifies the value for setpoint 8

P6	F9	RW	NRAM	0.000	<b>Setpoint 6</b> specifies the value for setpoint 8
P7	F9	RW	NRAM	0.000	<b>Setpoint 7</b> specifies the value for setpoint 8
P8	F9	RW	NRAM	0.000	<b>Setpoint 8</b> specifies the value for setpoint 8
LD	FE	RW	NRAM	300	<b>Loading Interval</b> specifies the loading interval in seconds
UD	FE	RW	NRAM	300	<b>Unloading Interval</b> specifies the unloading interval in seconds
LR	FE	RW	NRAM	0	<b>Time Until Next Loading Event Could Occur</b> specifies the time until the next loading event could occur in seconds
UR	FE	RW	NRAM	0	<b>Time Until Next Unloading Event Could Occur</b> specifies the time until the next loading event could occur in seconds
PR	FE	RW	NRAM	0	<b>Present Stages of Loading</b> indicates the amount of stages that are currently enabled
S1	FE	RO	NRAM	0	<b>Stage 1 Status</b> status indication for this stage
S2	FE	RO	NRAM	0	<b>Stage 2 Status</b> status indication for this stage
S3	FE	RO	NRAM	0	<b>Stage 3 Status</b> status indication for this stage
S4	FE	RO	NRAM	0	<b>Stage 4 Status</b> status indication for this stage
S5	FE	RO	NRAM	0	<b>Stage 5 Status</b> status indication for this stage
S6	FE	RO	NRAM	0	<b>Stage 6 Status</b> status indication for this stage
S7	FE	RO	NRAM	0	<b>Stage 7 Status</b> status indication for this stage
S8	FE	RO	NRAM	0	<b>Stage 8 Status</b> status indication for this stage
CI	DF	RW	NRAM		<b>Interlock Channel</b> specifies the channel of the input to be used for interlocking
IP	FE	RW	NRAM		<b>Interlock Polarity</b> specifies the polarity to be used for the interlocking input  0=Trigger on Non-Zero 1=Trigger on Zero
LP	FE	RW	NRAM		<b>Interlock Map</b> specifies the stages associated with the interlock
LV	E0	RO	NRAM	-1	<b>Interlock Input's Current Value</b> current value of the associated interlocking input

<b>IF</b>	FE	RO	NRAM	3	<b>Interlock Feedback</b> provides feedback based on interlocking  0=Unable to Read Input 1=Not Interlocked 2=Currently Interlocked 3=Interlocks Not In Use
<b>R1</b>	F6	RO	NRAM	0.000	<b>Stage 1 Run Hours</b> run hours count for the associated stage
<b>R2</b>	F6	RO	NRAM	0.000	<b>Stage2 Run Hours</b> run hours count for the associated stage
<b>R3</b>	F6	RO	NRAM	0.000	<b>Stage 3 Run Hours</b> run hours count for the associated stage
<b>R4</b>	F6	RO	NRAM	0.000	<b>Stage 4 Run Hours</b> run hours count for the associated stage
<b>R5</b>	F6	RO	NRAM	0.000	<b>Stage 5 Run Hours</b> run hours count for the associated stage
<b>R6</b>	F6	RO	NRAM	0.000	<b>Stage 6 Run Hours</b> run hours count for the associated stage
<b>R7</b>	F6	RO	NRAM	0.000	<b>Stage 7 Run Hours</b> run hours count for the associated stage
<b>R8</b>	F6	RO	NRAM	0.000	<b>Stage 8 Run Hours</b> run hours count for the associated stage
<b>CR</b>	FE	RW	NRAM	0	<b>Reset All Run Hours</b> determines if all of the run hours need to be cleared at once
<b>FB</b>	FE	RO	NRAM	0	<b>Staging Feedback</b> provides feedback for the staging channel  0=Operating Properly (Or Object Off) 1=Can't Read the Input 2=Setpoints are Identical 3=Setpoints are Reversed

MATH 1-15, F311-F31F

Attribute	Data Type	Access	Storage	Default	Description
<b>A1</b>	DF	RW	NRAM	0	<b>Input Attribute 1</b> specifies the attribute associated with the first input channel.
<b>A2</b>	DF	RW	NRAM	0	<b>Input Attribute 2</b> specifies the attribute associated with the second input channel.
<b>CV</b>	FE	RW	RAM	0	<b>Current Value</b> indicates the numeric result of applying the operand specified in <b>OP</b> to the inputs.
<b>DT</b>	FE	RW	NRAM	0	<b>PUP Data Type for Values</b> specifies the PUP datatype for <b>CV</b> .
<b>ET</b>	FE	RO	NRAM	0	<b>Input 1 is = To Input 2</b> provides feedback on whether Input 1 is equal to Input 2
<b>GE</b>	FE	RO	NRAM	0	<b>Input 1 is &gt;= Input 2</b> provides feedback on whether Input 1 is greater than or equal to Input 2
<b>GT</b>	FE	RO	NRAM	0	<b>Input 1 is &gt; Input 2</b> provides feedback on whether Input 1 is greater than Input 2
<b>I1</b>	DF	RW	NRAM	0	<b>Input channel 1</b> specifies the first input channel.
<b>I2</b>	DF	RW	NRAM	0	<b>Input channel 2</b> specifies the second input channel.
<b>LE</b>	FE	RO	NRAM	0	<b>Input 1 is &lt;= Input 2</b> specifies whether Input 1 is less than or equal to Input 2
<b>LT</b>	FE	RO	NRAM	0	<b>Input 1 is &lt; Input 2</b> specifies whether Input 1 is less than Input 2
<b>ON</b>	FC	RW	NRAM	"Math n" where n=1 to 15	<b>Math Channel Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>OP</b>	FE	RW	NRAM	0	<b>Operation</b> specifies the operation to be performed on the selected channels.  0=Disabled 1=Addition 2=Subtraction 3=Multiplication 4=Division
<b>V1</b>	FD	RO	NRAM	0	<b>Input 1's Value</b> specifies the value of Input 1
<b>V2</b>	FD	RO	NRAM	0	<b>Input 2's Value</b> specifies the value of Input 2

## AVERAGES 1-15, F301-F30F

Attribute	Data Type	Access	Storage	Default	Description
<b>A1</b>	DF	RW	NRAM	-	<b>Input Attribute 1</b> specifies the attribute associated with the first input channel.
<b>A2</b>	DF	RW	NRAM	-	<b>Input Attribute 2</b> specifies the attribute associated with the second input channel.
<b>A3</b>	DF	RW	NRAM	-	<b>Input Attribute 3</b> specifies the attribute associated with the third input channel.
<b>A4</b>	DF	RW	NRAM	-	<b>Input Attribute 4</b> specifies the attribute associated with the fourth input channel.
<b>AV</b>	FC	RW	RAM	0	<b>Average Value</b> displays the arithmetic mean of the inputs selected in <b>I1;A1</b> through <b>I4;A4</b> .
<b>CE</b>	FE	RW	NRAM	0	<b>Enable Calculations</b> enables and disables calculations associated with this channel
<b>DT</b>	FE	RW	NRAM	253	<b>PUP Datatype for Values</b> specifies the PUP datatype for <b>HV</b> , <b>LV</b> , and <b>AV</b> .
<b>HV</b>	FC	RW	NRAM	0	<b>High Value</b> indicated the maximum value of the inputs specified in <b>I1;A1</b> through <b>I4;A4</b> .
<b>I1</b>	DF	RW	NRAM	-	<b>Input channel 1</b> specifies the first input channel.
<b>I2</b>	DF	RW	NRAM	-	<b>Input channel 2</b> specifies the second input channel.
<b>I3</b>	DF	RW	NRAM	-	<b>Input channel 3</b> specifies the third input channel.
<b>I4</b>	DF	RW	NRAM	-	<b>Input channel 4</b> specifies the fourth input channel.
<b>LV</b>	FC	RW	NRAM	0	<b>Low Value</b> indicates the minimum value of the channels specified in <b>I1;A1</b> through <b>I4;A4</b> .
<b>ON</b>	DF	RW	NRAM	"Min/Max/Avg n" where n=1 to 15	<b>Average Channel Name</b> specifies a user definable string that is used to help identify the channel or its function.

LOGIC 1-16, F321-F330

Attribute	Data Type	Access	Storage	Default	Description
A1	DF	RW	NRAM	0	<b>Input Attribute 1</b> specifies the attribute associated with the first input channel.
A2	DF	RW	NRAM	0	<b>Input Attribute 2</b> specifies the attribute associated with the second input channel.
A3	DF	RW	NRAM	0	<b>Input Attribute 3</b> specifies the attribute associated with the third input channel.
A4	DF	RW	NRAM	0	<b>Input Attribute 4</b> specifies the attribute associated with the fourth input channel.
A5	DF	RW	NRAM	0	<b>Input Attribute 5</b> specifies the attribute associated with the fifth input channel.
A6	DF	RW	NRAM	0	<b>Input Attribute 6</b> specifies the attribute associated with the sixth input channel.
A7	DF	RW	NRAM	0	<b>Input Attribute 7</b> specifies the attribute associated with the seventh input channel.
A8	DF	RW	NRAM	0	<b>Input Attribute 8</b> specifies the attribute associated with the eighth input channel.
CV	FE	RO	RAM	0	<b>Current Value</b> indicates the result after the operand specified in <b>OP</b> has been applied to the inputs.
DT	FE	RW	NRAM	253	<b>PUP Datatype for Values</b> specifies the PUP datatype for <b>CV</b> .
I1	DF	RW	NRAM	0	<b>Input channel 1</b> specifies the first input channel.
I2	DF	RW	NRAM	0	<b>Input channel 2</b> specifies the second input channel.
I3	DF	RW	NRAM	0	<b>Input channel 3</b> specifies the third input channel.
I4	DF	RW	NRAM	0	<b>Input channel 4</b> specifies the fourth input channel.
I5	DF	RW	NRAM	0	<b>Input channel 5</b> specifies the fifth input channel.
I6	DF	RW	NRAM	0	<b>Input channel 6</b> specifies the sixth input channel.
I7	DF	RW	NRAM	0	<b>Input channel 7</b> specifies the seventh input channel.
I8	DF	RW	NRAM	0	<b>Input channel 8</b> specifies the eighth input channel.
ON	DF	RW	NRAM	"Logic n" where n=1 to 16	<b>Logic Channel Name</b> specifies a user definable string that is used to help identify the channel or its function.

Attribute	Data Type	Access	Storage	Default	Description
<b>OP</b>	FE	RW	NRAM	0	<p><b>Operation</b> specifies the logic operation to be performed on the selected channels.</p> <p>0=Disabled 1=OR 2=AND 3=NOT 4=XOR</p>
<b>V1</b>	FE	RO	NRAM	2	<p><b>Input 1's Value</b> specifies the current value of Input 1</p> <p>0=Off 1=On 2=Unreliable</p>
<b>V2</b>	FE	RO	NRAM	2	<p><b>Input 2's Value</b> specifies the current value of Input 2</p> <p>0=Off 1=On 2=Unreliable</p>
<b>V3</b>	FE	RO	NRAM	2	<p><b>Input 3's Value</b> specifies the current value of Input 3</p> <p>0=Off 1=On 2=Unreliable</p>
<b>V4</b>	FE	RO	NRAM	2	<p><b>Input 4's Value</b> specifies the current value of Input 4</p> <p>0=Off 1=On 2=Unreliable</p>
<b>V5</b>	FE	RO	NRAM	2	<p><b>Input 5's Value</b> specifies the current value of Input 5</p> <p>0=Off 1=On 2=Unreliable</p>
<b>V6</b>	FE	RO	NRAM	2	<p><b>Input 6's Value</b> specifies the current value of Input 6</p> <p>0=Off 1=On 2=Unreliable</p>
<b>V7</b>	FE	RO	NRAM	2	<p><b>Input 7's Value</b> specifies the current value of Input 7</p> <p>0=Off 1=On 2=Unreliable</p>
<b>V8</b>	FE	RO	NRAM	2	<p><b>Input 8's Value</b> specifies the current value of Input 8</p> <p>0=Off 1=On 2=Unreliable</p>

PROGRAM SUMMARY, F200

Attribute	Data Type	Access	Storage	Default	Description
\$1	FE	RW	NRAM	0	<p><b>Program 1 Status</b>  indicates the current status of the resident SPL program number 1.</p> <p>0=Stop  1=Run  2=Unloaded  3=Abort  4=Wait for Time  5=Restart  6=Load  7=Unload Request  8=Abort Request  9=Wait for Fetch</p>
\$2	FE	RW	NRAM	0	<p><b>Program 2 Status</b>  indicates the current status of the resident SPL program number 2.</p> <p>0=Stop  1=Run  2=Unloaded  3=Abort  4=Wait for Time  5=Restart  6=Load  7=Unload Request  8=Abort Request  9=Wait for Fetch</p>
\$3	FE	RW	NRAM	0	<p><b>Program 3 Status</b>  indicates the current status of the resident SPL program number 3.</p> <p>0=Stop  1=Run  2=Unloaded  3=Abort  4=Wait for Time  5=Restart  6=Load  7=Unload Request  8=Abort Request  9=Wait for Fetch</p>
\$4	FE	RW	NRAM	0	<p><b>Program 4 Status</b>  indicates the current status of the resident SPL program number 4.</p> <p>0=Stop  1=Run  2=Unloaded  3=Abort  4=Wait for Time  5=Restart  6=Load  7=Unload Request  8=Abort Request  9=Wait for Fetch</p>
\$5	FE	RW	NRAM	0	<p><b>Program 5 Status</b>  indicates the current status of the resident SPL program number 6.</p> <p>0=Stop  1=Run  2=Unloaded  3=Abort  4=Wait for Time  5=Restart  6=Load  7=Unload Request  8=Abort Request  9=Wait for Fetch</p>

Attribute	Data Type	Access	Storage	Default	Description
\$6	FE	RW	NRAM	0	<p><b>Program 6 Status</b> indicates the current status of the resident SPL program number 6.</p> <p>0=Stop 1=Run 2=Unloaded 3=Abort 4=Wait for Time 5=Restart 6=Load 7=Unload Request 8=Abort Request 9=Wait for Fetch</p>
\$7	FE	RW	NRAM	0	<p><b>Program 7 Status</b> indicates the current status of the resident SPL program number 7.</p> <p>0=Stop 1=Run 2=Unloaded 3=Abort 4=Wait for Time 5=Restart 6=Load 7=Unload Request 8=Abort Request 9=Wait for Fetch</p>
\$8	FE	RW	NRAM	0	<p><b>Program 8 Status</b> indicates the current status of the resident SPL program number 8.</p> <p>0=Stop 1=Run 2=Unloaded 3=Abort 4=Wait for Time 5=Restart 6=Load 7=Unload Request 8=Abort Request 9=Wait for Fetch</p>
ON	DF	RW	NRAM	"Program Summary"	<p><b>Program Summary Name</b> specifies a user definable string that is used to help identify the channel or its function.</p>

PROGRAMS 1-8, F201-F208

Attribute	Data Type	Access	Storage	Default	Description
<b>\$\$</b>	FE	RW	NRAM	0	<p><b>Current Status</b> indicates the current status of the resident SPL program.</p> <p>0=Stop 1=Run 2=Unloaded 3=Abort 4=Wait for Time 5=Restart 6=Load 7=Unload Request 8=Abort Request 9=Wait for Fetch</p>
<b>\$1</b>	FE	RW	NRAM	0	<p><b>Enable Single-Step Mode?</b> specifies whether the single-step, line by line debugging mode is enabled.</p> <p>0=No 1=Yes</p>
<b>\$C</b>	FE	RO	NRAM	0	<p><b>Program Counter</b> indicates the hexadecimal memory location of the next program statement to be executed.</p>
<b>\$D</b>	FE	RW	NRAM	0	<p><b>Delay Time Remaining</b> specifies the number of seconds remaining when an <b>SWAIT</b> or <b>MWAIT</b> statement is encountered in the SPL program.</p>
<b>\$E</b>	FE	RW	NRAM	0	<p><b>Error Code</b> indicates the SPL error code that is returned when the program aborts.</p>
<b>\$S</b>	FE	RW	NRAM	0	<p><b>Section Number</b> indicates the current section number as determined by the <b>SECTION</b> statements in the SPL code.</p>
<b>\$W</b>	FE	RO	NRAM	0	<p><b>Trappable Error Action</b> specifies how the SPL program should handle trappable errors.</p> <p>0=Abort on Error 1=Wait on Error</p>
<b>%A</b>	FE	RW	NRAM	0	<p><b>Register A Value</b> indicates the value of program register A.</p>
<b>%B</b>	FE	RW	NRAM	0	<p><b>Register B Value</b> indicates the value of program register B.</p>
<b>%C</b>	FE	RW	NRAM	0	<p><b>Register C Value</b> indicates the value of program register C.</p>
<b>%D</b>	FE	RW	NRAM	0	<p><b>Register D Value</b> indicates the value of program register A.</p>
<b>%E</b>	FE	RW	NRAM	0	<p><b>Register E Value</b> indicates the value of program register E.</p>
<b>%F</b>	FE	RW	NRAM	0	<p><b>Register F Value</b> indicates the value of program register F.</p>
<b>%G</b>	FE	RW	NRAM	0	<p><b>Register G Value</b> indicates the value of program register G.</p>

Attribute	Data Type	Access	Storage	Default	Description
<b>%H</b>	FE	RW	NRAM	0	<b>Register H Value</b> indicates the value of program register H.
<b>%I</b>	FE	RW	NRAM	0	<b>Register I Value</b> indicates the value of program register I.
<b>%J</b>	FE	RW	NRAM	0	<b>Register J Value</b> indicates the value of program register J.
<b>%K</b>	FE	RW	NRAM	0	<b>Register K Value</b> indicates the value of program register K.
<b>%L</b>	FE	RW	NRAM	0	<b>Register L Value</b> indicates the value of program register L.
<b>%M</b>	FE	RW	NRAM	0	<b>Register M Value</b> indicates the value of program register M.
<b>%N</b>	FE	RW	NRAM	0	<b>Register N Value</b> indicates the value of program register N.
<b>%O</b>	FE	RW	NRAM	0	<b>Register O Value</b> indicates the value of program register O.
<b>%P</b>	FE	RW	NRAM	0	<b>Register P Value</b> indicates the value of program register P.
<b>ET</b>	FE	RW	NRAM		<b>Error Text</b> provides text feedback on the abort cause
<b>ON</b>	DF	RW	NRAM	"Program n" where n=1 to 8	<b>Program Name</b> specifies a user definable string that is used to help identify the channel or its function.

INPUT SELECT 1-15, F011-F01F

Attribute	Data Type	Access	Storage	Default	Description
<b>A1</b>	DF	RW	NRAM	0	<b>Input Attribute 1</b> specifies the attribute associated with the first input channel.
<b>A2</b>	DF	RW	NRAM	0	<b>Input Attribute 2</b> specifies the attribute associated with the second input channel.
<b>CV</b>	FE	RO	RAM	0	<b>Current Value</b>
<b>DT</b>	FE	RW	NRAM	0	<b>PUP Data Type for Values</b> specifies the PUP datatype for <b>CV</b> .
<b>FB</b>	FE	RO	NRAM	1	<b>Feedback</b> provides feedback for this channel  0=Working Properly 1=Unable to read inputs
<b>I1</b>	DF	RW	NRAM		<b>Input channel 1</b> specifies the first input channel.
<b>I2</b>	DF	RW	NRAM		<b>Input channel 2</b> specifies the second input channel.
<b>ON</b>	DF	RW	NRAM	"Input Select n" where n=1 to 15	<b>Input Select Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>SA</b>	FE	RW	NRAM	0	<b>Selection Attribute</b> specifies the attribute associated with the selection channel used as the selection criteria.
<b>SC</b>	FE	RW	NRAM	0	<b>Selection Channel</b> specifies the channel used as the selection criteria.
<b>SV</b>	FE	RW	NRAM	0	<b>Select Input's Value</b> specifies which input should be used for current value  0=Input 1 1=Input 2

**BROADCAST 0, F000**

Attribute	Data Type	Access	Storage	Default	Description
<b>BM</b>	FE	RW	NRAM	0	<b>Broadcast Mode</b> indicates the current mode for the broadcast channel  0=Off 1=Send 2=Receive
<b>BT</b>	FE	RW	NRAM	5	<b>Minutes Between Broadcasts</b> indicates the amount of time in minutes between broadcasts
<b>BZ</b>	FE	RW	NRAM	1	<b>Broadcast Zone/Global</b> specifies whether the controller's broadcasts are sent to the zone or broadcast globally.  0=Zone broadcast 1=Global broadcast
<b>CV</b>	FB	RW	RAM		<b>Current Value</b> indicates the current value of the referenced channel specified in IC.
<b>DT</b>	FE	RW	NRAM	253	<b>PUP Data Type for Values</b> specifies the PUP datatype for the broadcast.
<b>ES</b>	FE	RO	NRAM	65535	<b>Elapsed Seconds Since Broadcast</b> indicates the amount of time in seconds since the last broadcast
<b>FB</b>	FE	RO	NRAM	2	<b>Feedback Status</b> provides feedback for the associated broadcast channel  0=Set to Receive 1=Unable to Read Input 2=Working Properly 3=Unable to Coerce Input Datatype
<b>IA</b>	DF	RW	NRAM	0	<b>Input Attribute</b> specifies the attribute associated with the channel specified in IC to be broadcast.
<b>IC</b>	DF	RW	NRAM		<b>Input Channel</b> specifies the input channel to be broadcast.
<b>ON</b>	DF	RW	NRAM	Broadcast 0	<b>Broadcast Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>ZN</b>	FE	RW	NRAM	0	<b>Zone Number</b> specifies the zone number the broadcast channel will be using for its broadcast mode

BROADCAST 1, F001

Attribute	Data Type	Access	Storage	Default	Description
<b>BM</b>	FE	RW	NRAM	0	<b>Broadcast Mode</b> indicates the current mode for the broadcast channel  0=Off 1=Send 2=Receive
<b>BT</b>	FE	RW	NRAM	5	<b>Minutes Between Broadcasts</b> indicates the amount of time in minutes between broadcasts
<b>BZ</b>	FE	RW	NRAM	1	<b>Broadcast Zone/Global</b> specifies whether the controller's broadcasts are sent to the zone or broadcast globally.  0=Zone broadcast 1=Global broadcast
<b>CV</b>	FB	RW	RAM		<b>Current Value</b> indicates the current value of the referenced channel specified in IC.
<b>DT</b>	FE	RW	NRAM	253	<b>PUP Data Type for Values</b> specifies the PUP datatype for the broadcast.
<b>ES</b>	FE	RO	NRAM	65535	<b>Elapsed Seconds Since Broadcast</b> indicates the amount of time in seconds since the last broadcast
<b>FB</b>	FE	RO	NRAM	2	<b>Feedback Status</b> provides feedback for the associated broadcast channel  0=Set to Receive 1=Unable to Read Input 2=Working Properly 3=Unable to Coerce Input Datatype
<b>IA</b>	DF	RW	NRAM	0	<b>Input Attribute</b> specifies the attribute associated with the channel specified in IC to be broadcast.
<b>IC</b>	DF	RW	NRAM		<b>Input Channel</b> specifies the input channel to be broadcast.
<b>ON</b>	DF	RW	NRAM	Broadcast 1	<b>Broadcast Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>ZN</b>	FE	RW	NRAM	0	<b>Zone Number</b> specifies the zone number the broadcast channel will be using for its broadcast mode
<b>ON</b>	DF	RW	NRAM	Broadcast 1	<b>Broadcast Name</b> specifies a user definable string that is used to help identify the channel or its function.

## BROADCAST 2, F002

Attribute	Data Type	Access	Storage	Default	Description
<b>BM</b>	FE	RW	NRAM	0	<b>Broadcast Mode</b> indicates the current mode for the broadcast channel  0=Off 1=Send 2=Receive
<b>BT</b>	FE	RW	NRAM	5	<b>Minutes Between Broadcasts</b> indicates the amount of time in minutes between broadcasts
<b>BZ</b>	FE	RW	NRAM	1	<b>Broadcast Zone/Global</b> specifies whether the controller's broadcasts are sent to the zone or broadcast globally.  0=Zone broadcast 1=Global broadcast
<b>CV</b>	FB	RW	RAM		<b>Current Value</b> indicates the current value of the referenced channel specified in IC.
<b>DT</b>	FE	RW	NRAM	253	<b>PUP Data Type for Values</b> specifies the PUP datatype for the broadcast.
<b>ES</b>	FE	RO	NRAM	65535	<b>Elapsed Seconds Since Broadcast</b> indicates the amount of time in seconds since the last broadcast
<b>FB</b>	FE	RO	NRAM	2	<b>Feedback Status</b> provides feedback for the associated broadcast channel  0=Set to Receive 1=Unable to Read Input 2=Working Properly 3=Unable to Coerce Input Datatype
<b>IA</b>	DF	RW	NRAM	0	<b>Input Attribute</b> specifies the attribute associated with the channel specified in IC to be broadcast.
<b>IC</b>	DF	RW	NRAM		<b>Input Channel</b> specifies the input channel to be broadcast.
<b>ON</b>	DF	RW	NRAM	Broadcast 2	<b>Broadcast Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>ZN</b>	FE	RW	NRAM	0	<b>Zone Number</b> specifies the zone number the broadcast channel will be using for its broadcast mode

BROADCAST 3, F003

Attribute	Data Type	Access	Storage	Default	Description
<b>BM</b>	FE	RW	NRAM	0	<b>Broadcast Mode</b> indicates the current mode for the broadcast channel  0=Off 1=Send 2=Receive
<b>BT</b>	FE	RW	NRAM	5	<b>Minutes Between Broadcasts</b> indicates the amount of time in minutes between broadcasts
<b>BZ</b>	FE	RW	NRAM	1	<b>Broadcast Zone/Global</b> specifies whether the controller's broadcasts are sent to the zone or broadcast globally.  0=Zone broadcast 1=Global broadcast
<b>CV</b>	FB	RW	RAM		<b>Current Value</b> indicates the current value of the referenced channel specified in IC.
<b>DT</b>	FE	RW	NRAM	253	<b>PUP Data Type for Values</b> specifies the PUP datatype for the broadcast.
<b>ES</b>	FE	RO	NRAM	65535	<b>Elapsed Seconds Since Broadcast</b> indicates the amount of time in seconds since the last broadcast
<b>FB</b>	FE	RO	NRAM	2	<b>Feedback Status</b> provides feedback for the associated broadcast channel  0=Set to Receive 1=Unable to Read Input 2=Working Properly 3=Unable to Coerce Input Datatype
<b>IA</b>	DF	RW	NRAM	0	<b>Input Attribute</b> specifies the attribute associated with the channel specified in IC to be broadcast.
<b>IC</b>	DF	RW	NRAM		<b>Input Channel</b> specifies the input channel to be broadcast.
<b>ON</b>	DF	RW	NRAM	Broadcast 3	<b>Broadcast Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>ZN</b>	FE	RW	NRAM	0	<b>Zone Number</b> specifies the zone number the broadcast channel will be using for its broadcast mode

## BROADCAST 4, F004

Attribute	Data Type	Access	Storage	Default	Description
<b>BM</b>	FE	RW	NRAM	0	<b>Broadcast Mode</b> indicates the current mode for the broadcast channel  0=Off 1=Send 2=Receive
<b>BT</b>	FE	RW	NRAM	5	<b>Minutes Between Broadcasts</b> indicates the amount of time in minutes between broadcasts
<b>BZ</b>	FE	RW	NRAM	1	<b>Broadcast Zone/Global</b> specifies whether the controller's broadcasts are sent to the zone or broadcast globally.  0=Zone broadcast 1=Global broadcast
<b>CV</b>	FB	RW	RAM		<b>Current Value</b> indicates the current value of the referenced channel specified in IC.
<b>DT</b>	FE	RW	NRAM	253	<b>PUP Data Type for Values</b> specifies the PUP datatype for the broadcast.
<b>ES</b>	FE	RO	NRAM	65535	<b>Elapsed Seconds Since Broadcast</b> indicates the amount of time in seconds since the last broadcast
<b>FB</b>	FE	RO	NRAM	2	<b>Feedback Status</b> provides feedback for the associated broadcast channel  0=Set to Receive 1=Unable to Read Input 2=Working Properly 3=Unable to Coerce Input Datatype
<b>IA</b>	DF	RW	NRAM	0	<b>Input Attribute</b> specifies the attribute associated with the channel specified in IC to be broadcast.
<b>IC</b>	DF	RW	NRAM		<b>Input Channel</b> specifies the input channel to be broadcast.
<b>ON</b>	DF	RW	NRAM	Broadcast 4	<b>Broadcast Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>ZN</b>	FE	RW	NRAM	0	<b>Zone Number</b> specifies the zone number the broadcast channel will be using for its broadcast mode

BROADCAST 5, F005

Attribute	Data Type	Access	Storage	Default	Description
<b>BM</b>	FE	RW	NRAM	0	<b>Broadcast Mode</b> indicates the current mode for the broadcast channel  0=Off 1=Send 2=Receive
<b>BT</b>	FE	RW	NRAM	5	<b>Minutes Between Broadcasts</b> indicates the amount of time in minutes between broadcasts
<b>BZ</b>	FE	RW	NRAM	1	<b>Broadcast Zone/Global</b> specifies whether the controller's broadcasts are sent to the zone or broadcast globally.  0=Zone broadcast 1=Global broadcast
<b>CV</b>	FB	RW	RAM		<b>Current Value</b> indicates the current value of the referenced channel specified in IC.
<b>DT</b>	FE	RW	NRAM	253	<b>PUP Data Type for Values</b> specifies the PUP datatype for the broadcast.
<b>ES</b>	FE	RO	NRAM	65535	<b>Elapsed Seconds Since Broadcast</b> indicates the amount of time in seconds since the last broadcast
<b>FB</b>	FE	RO	NRAM	2	<b>Feedback Status</b> provides feedback for the associated broadcast channel  0=Set to Receive 1=Unable to Read Input 2=Working Properly 3=Unable to Coerce Input Datatype
<b>IA</b>	DF	RW	NRAM	0	<b>Input Attribute</b> specifies the attribute associated with the channel specified in IC to be broadcast.
<b>IC</b>	DF	RW	NRAM		<b>Input Channel</b> specifies the input channel to be broadcast.
<b>ON</b>	DF	RW	NRAM	Broadcast 5	<b>Broadcast Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>ZN</b>	FE	RW	NRAM	0	<b>Zone Number</b> specifies the zone number the broadcast channel will be using for its broadcast mode

## BROADCAST 6, F006

Attribute	Data Type	Access	Storage	Default	Description
<b>BM</b>	FE	RW	NRAM	0	<b>Broadcast Mode</b> indicates the current mode for the broadcast channel  0=Off 1=Send 2=Receive
<b>BT</b>	FE	RW	NRAM	5	<b>Minutes Between Broadcasts</b> indicates the amount of time in minutes between broadcasts
<b>BZ</b>	FE	RW	NRAM	1	<b>Broadcast Zone/Global</b> specifies whether the controller's broadcasts are sent to the zone or broadcast globally.  0=Zone broadcast 1=Global broadcast
<b>CV</b>	FB	RW	RAM		<b>Current Value</b> indicates the current value of the referenced channel specified in IC.
<b>DT</b>	FE	RW	NRAM	253	<b>PUP Data Type for Values</b> specifies the PUP datatype for the broadcast.
<b>ES</b>	FE	RO	NRAM	65535	<b>Elapsed Seconds Since Broadcast</b> indicates the amount of time in seconds since the last broadcast
<b>FB</b>	FE	RO	NRAM	2	<b>Feedback Status</b> provides feedback for the associated broadcast channel  0=Set to Receive 1=Unable to Read Input 2=Working Properly 3=Unable to Coerce Input Datatype
<b>IA</b>	DF	RW	NRAM	0	<b>Input Attribute</b> specifies the attribute associated with the channel specified in IC to be broadcast.
<b>IC</b>	DF	RW	NRAM		<b>Input Channel</b> specifies the input channel to be broadcast.
<b>ON</b>	DF	RW	NRAM	Broadcast 6	<b>Broadcast Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>ZN</b>	FE	RW	NRAM	0	<b>Zone Number</b> specifies the zone number the broadcast channel will be using for its broadcast mode

BROADCAST 7, F007

Attribute	Data Type	Access	Storage	Default	Description
<b>BM</b>	FE	RW	NRAM	0	<b>Broadcast Mode</b> indicates the current mode for the broadcast channel  0=Off 1=Send 2=Receive
<b>BT</b>	FE	RW	NRAM	5	<b>Minutes Between Broadcasts</b> indicates the amount of time in minutes between broadcasts
<b>BZ</b>	FE	RW	NRAM	1	<b>Broadcast Zone/Global</b> specifies whether the controller's broadcasts are sent to the zone or broadcast globally.  0=Zone broadcast 1=Global broadcast
<b>CV</b>	FB	RW	RAM		<b>Current Value</b> indicates the current value of the referenced channel specified in IC.
<b>DT</b>	FE	RW	NRAM	253	<b>PUP Data Type for Values</b> specifies the PUP datatype for the broadcast.
<b>ES</b>	FE	RO	NRAM	65535	<b>Elapsed Seconds Since Broadcast</b> indicates the amount of time in seconds since the last broadcast
<b>FB</b>	FE	RO	NRAM	2	<b>Feedback Status</b> provides feedback for the associated broadcast channel  0=Set to Receive 1=Unable to Read Input 2=Working Properly 3=Unable to Coerce Input Datatype
<b>IA</b>	DF	RW	NRAM	0	<b>Input Attribute</b> specifies the attribute associated with the channel specified in IC to be broadcast.
<b>IC</b>	DF	RW	NRAM		<b>Input Channel</b> specifies the input channel to be broadcast.
<b>ON</b>	DF	RW	NRAM	Broadcast 7	<b>Broadcast Name</b> specifies a user definable string that is used to help identify the channel or its function.
<b>ZN</b>	FE	RW	NRAM	0	<b>Zone Number</b> specifies the zone number the broadcast channel will be using for its broadcast mode

## SPECIAL MODES SUMMARY E000

Attribute	Data Type	Access	Storage	Default	Description
<b>FA</b>	FA	RO	NRAM		<b>Faults Detected</b> indicates that the controller has detected a fire or communication failure fault

FIRE MODE E001

Attribute	Data Type	Access	Storage	Default	Description
<b>BF</b>	FE	RW	NRAM	0	<b>Broadcast Fire Mode (When Entering Only)</b> indicates whether fire mode will be broadcast  0=No 1=Yes
<b>CV</b>	FE	RO	RAM	0	<b>Current Value</b> indicates the current value of fire mode  0=Normal 1=Fire Mode
<b>FB</b>	FE	RO	NRAM		<b>Fire Mode Feedback</b> provides feedback on fire mode  0=Not in Fire Mode 1=Locked in By Local Input 2=Waiting for Timer to Expire 3=Waiting to be Manually Cleared
<b>FM</b>	FE	RO	NRAM		<b>Time Remaining in Fire Mode</b> specifies the time remaining for fire mode in seconds
<b>MC</b>	FE	RW	NRAM		<b>Manually Clear Fire Mode</b> specifies whether fire mode should be manually cleared
<b>TF</b>	FE	RW	NRAM	45	<b>Time to Remain in Fire Mode</b> specifies the time to remain in fire mode in seconds

## COMM FAILURE E002

Attribute	Data Type	Access	Storage	Default	Description
<b>BD</b>	FE	RW	NRAM	20	<b>Failure Mode Boot Delay</b> specifies the amount of time in seconds before communication failure can be indicated after the controller boots up
<b>CV</b>	FE	RO	RAM	0	<b>Current Value</b> indicates the current value for comm failure  0=Comm Failure 1=Good
<b>FD</b>	FE	RW	NRAM	10	<b>Failure Mode Delay Time</b> specifies the delay in seconds before communications failure is indicated
<b>FM</b>	FE	RO	NRAM	0	<b>Failure Mode</b> specifies when to mark comm failure  0=Always Marked as Good 1=Fail if not passed a Token 2=Fail if no data is read/written
<b>OS</b>	FE	RW	NRAM	0	<b>Out of Service</b> Allows channel to be overwritten  0=False 1=True



## APPENDIX B: SPL ERROR CODES

This section lists the possible error codes that can be generated by a running SPL program.

Code	Trappable	Meaning	Possible Causes/Examples
1	x	<b>Stack Overflow</b>	A value larger than the largest value supported by program executor was encountered.
2	x	<b>Stack Underflow</b>	A value smaller than the smallest value supported by the program executor was encountered.
3	x	<b>Invalid Format String</b>	The .plb file is corrupted. The line of SPL is invalid.
4	x	<b>Invalid Coercion</b>	An attempt was made to coerce a variable to an incompatible datatype
5	x	<b>Expression Stack Overflow</b>	Internal SPL error.
6		<b>Expression Stack Underflow</b>	Internal SPL error.
7	x	<b>Invalid Expression State</b>	Internal SPL error
8	x	<b>Invalid PCode</b>	The function or statement is invalid. Either the function is unsupported or the statement was used incorrectly.
9	x	<b>Invalid Term</b>	A reference or attribute has been misused.
10	x	<b>Not Implemented</b>	The function or statement is not implemented in the controller.
11	x	<b>On Goto</b>	The GOTO statement specified a target which is out of range.
12	x	<b>Bad Reference</b>	A reference in the source code is out of range.
13	x	<b>Invalid Datatype</b>	The referenced datatype does not match the datatype being written.
14	x	<b>Format Mismatch</b>	The .plb file is corrupted. The SPL expression does not match the required parameter format.
15	x	<b>Invalid Operator</b>	An unsupported operator was encountered during the execution of the program code.
16	x	<b>Table Read Only</b>	An attempt has been made to write to a table which is read only.
17	x	<b>Nesting Overflow</b>	The maximum number of nested expressions has been exceeded.

Code	Trappable	Meaning	Possible Causes/Examples
18	x	<b>Queue Full</b>	The network transmission has been denied due to the transmission queue being full.

## APPENDIX C: PUP DATA TYPES

*This Appendix lists the hexadecimal and decimal PUP numeric data type codes. The hexadecimal codes are followed by h and the decimal codes are provided in parentheses.*

Code	Digit Format	Meaning
FFh (255)	±XXXXXXXXXX.	signed 10 digit
FEh (254)	XXXXXXXXXX.	unsigned 10 digit
FDh (253)	±XXXXXXXXX.X	signed 9.1 digit
FCh (252)	XXXXXXXXX.X	unsigned 9.1 digit
FBh (251)	±XXXXXXXXXX	signed 8.2 digit
FAh (250)	XXXXXXXXXX	unsigned 8.2 digit
F9h (249)	±XXXXXXX.XXX	signed 7.3 digit
F8h (248)	XXXXXXX.XXX	unsigned 7.3 digit
F7h (247)	±XXXXXX.XXXX	signed 6.4 digit
F6h (246)	XXXXXX.XXXX	unsigned 6.4 digit
F5h (245)	±XXXXX.XXXXX	signed 5.5 digit
F4h (244)	XXXXX.XXXXX	unsigned 5.5 digit
F3h (243)	±XXXX.XXXXXX	signed 4.6 digit
F2h (242)	XXXX.XXXXXX	unsigned 4.6 digit
F1h (241)	±XXX.XXXXXXX	signed 3.7 digit
F0h (240)	XXX.XXXXXXX	unsigned 3.7 digit
EFh (239)	±XX.XXXXXXXXX	signed 2.8 digit
EEh (238)	XX.XXXXXXXXX	unsigned 2.8 digit
EDh (237)	±X.XXXXXXXXXX	signed 1.9 digit
ECh (236)	X.XXXXXXXXXX	unsigned 1.9 digit
EBh (235)	±.XXXXXXXXXXX	signed .10 digit
EAh (234)	.XXXXXXXXXXX	unsigned .10 digit
E9h (233)	channel map	one bit per channel

---

Code	Digit Format	Meaning
E8h (232)	bitmap of text	one bit per text field
E7h (231)	BCD (H/S/M)	hours is LSB
E6h (230)	BCD (H/M)	hours is LSB
E5h (229)	packed BCD	8 BCD digits as 4 bytes
E4h (228)	BCD date (Y/M/D)	MSW is year LSW/MSB is month LSW/LSB is day
E3h (227)	Binary date	MSW is year LSW/MSB is month LSW/LSB is day
E2h (226)	reserved	
E1h (225)	reserved	
E0h (224)	IEEE 754 32-bit floating point	
DFh-00h (223-0)	reserved	